НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
Інститут прикладного системного аналізу
Кафедра математичних методів системного аналізу

«До захисту допущено»
В. о. завідувача кафедри
_____ О.Л. Тимощук
«___»_____20__ р.

Дипломна робота

на здобуття ступеня бакалавра
з напряму підготовки 6.040303 «Системний аналіз»
на тему:
«Адаптивний метод прийняття рішень для ринку цінних паперів на основі ефектуаційної концепції»

Виконав:
студент IV курсу, групи КА-51
Канцедал Георгій Олегович _____

Керівник:
доцент, к.ф.-м.н. Каніовська І.Ю. _____

Консультант з економічного розділу:
доцент, к.е.н. Шевчук О.А. _____

Консультант з нормоконтролю:
доцент, к.т.н. Коваленко А.Є. _____

Рецензент:
доцент, к.ф.-м.н. Буценко Ю.П. _____

Засвідчую, що у цій дипломній роботі
немає запозичень з праць інших авторів
без відповідних посилань.
Студент _____

Київ – 2019 року

Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Інститут прикладного системного аналізу

Кафедра математичних методів системного аналізу

Рівень вищої освіти – перший (бакалаврський)

Напрям підготовки (програма професійного спрямування) – 6.040303

«Системний аналіз» («Системний аналіз і управління»)

ЗАТВЕРДЖУЮ

В.о.завідувача кафедри

_____ О.Л. Тимощук

«___»_____20__ р.

ЗАВДАННЯ

на дипломну роботу студенту

Канцедалу Георгію Олеговичу

1. Тема роботи «Адаптивний метод прийняття рішень для ринку цінних паперів на основі ефектуаційної концепції», керівник роботи Каніовська Ірина Юріївна, доцента кафедри математичних методів системного аналізу, к.ф.-м.н., доц., затверджена наказом по університету від «25»травня 2019 р. №1353с.

2. Термін подання студентом роботи _____

3. Вихідні дані до роботи _____

_____

4. Зміст роботи _____

_____

_____

5. Перелік ілюстративного матеріалу (із зазначенням плакатів, презентацій тощо)
_____

_____

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Економічний	Шевчук О.А., доцент		

7. Дата видачі завдання _____

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітка

Студент _____ _____
(підпис) (ініціали, прізвище)

Керівник роботи _____ _____
(підпис) (ініціали, прізвище)

РЕФЕРАТ

Дипломна робота: 78 с., 31 рис., 6 табл., 2 дод., 13 джерел.

АДАПТИВНИЙ МЕТОД ПРИЙНЯТТЯ РІШЕНЬ ДЛЯ РИНКУ ЦІННИХ ПАПЕРІВ НА ОСНОВІ ЕФЕКТУАЦІЙНОЇ КОНЦЕПЦІЇ, СИСТЕМИ ПРИЙНЯТТЯ РІШЕНЬ, РЕГУЛЯРИЗАЦІЯ БАЗ ДАНИХ, ГРАДІЄНТНИ БУСТИНГ

Об’єкт дослідження – Фінансові часові ряди цінних паперів, історія торгівельних операцій.

Предмет дослідження – Методи класифікації, що базуються на машинному навчанні.

Мета роботи – Проаналізувати предмет дослідження, реалізувати безперервний доступ до трейдингових даних, провести тестування побудованої моделі на історичних даних та в режимі реального часу.

Методи дослідження – методи машинного навчання: дерева прийняття рішень, випадковий ліс, нейроні мережі, градієнтний бустинг.

Актуальність – задача прийняття рішень на ринках цінних паперів є провідною для сучасного світу. Побудова стратегій з низьким ступенем ризику є оптимальним вибором для збереження пасивів будь-якого підприємства. Побудована модель може бути легко перенесена на будь-який ринок, в основі якого лежить часовий ряд та контрагентами ряду виступають рівноправні користувачі без значної переваги.

Результати роботи свідчать досягнення рентабельності використання активів при застосуванні моделі на рівні середньої банківської ставки 8% в місяць.

Шляхи подальшого розвитку предмету дослідження – збільшити кількості залучених параметрів активів в моделі; застосування стратегій для торгівлі іншими фінансовими інструментами та адаптації моделі для проведення операцій між декількома парами.

ABSTRACT

Diploma work: 78 p., 31 fig., 6 tabl., 2 appendixes, 13 references.

ALGORITHMIC TRADING, , TIME SERIES ANALYSIS, COINTEGRATION, STATIONARITY, ADAPTIVE DECISION SUPPORT SYSTEM, STOCKS MARKETS, EFFECTUATION THEORY

Object of research - Financial time series of securities, history of trading operations.

Subject of study - Classification methods based on machine learning.

The purpose of the work - To analyze the subject of research, to implement continuous access to trading data, to test the built model on historical data and in real time.

Methods of research - methods of machine learning: decision tree, random forest, neural network, gradient boosting.

Actuality - the task of making decisions on the securities markets is leading for the modern world. Building low risk strategies is the best choice for keeping any company's liabilities. The constructed model can easily be transferred to any market, which is based on the time series and counterparts of the series are equal users without significant advantage.

The results of the work indicate the achievement of the profitability of the use of assets in applying the model at the level of the average bank rate of 8% per month.

Ways of further development of the subject of research - to increase the number of attracted asset parameters in the model; use strategies for trading other financial instruments and adapt the model for multi-pair transactions.

ЗМІСТ

ВСТУП

У даній роботі досліджується ефектуаційний підхід до проведення торгівельних операцій на ринку цінних паперів. Проведено дослідження ринку Розкриття ефектуаційного підходу до проведення торгівельних операцій на ринку цінних паперів. Застосування методів машинного навчання на основі дерев прийняття рішень за для досягнення максимальної швидкості проведення торгівельних операцій. Розроблена концепція протестована в реальному часі на офіційних торгівельних даних, досягнуте за рахунок розробленого комплексу програмного забезпечення.

У першому розділі розглядаються базові принципи ефектуаційної концепції, та методи дослідження ринку, що були застосовані. Проведено огляд основних понять, методів.

Другий розділ присвячено впровадженню запропонованої моделі на основі попереднього аналізу ринку. Обґрунтовано вибір ринкової пари, розглянуто параметри моделі та описано їх вплив на її формування.

Третій розділ містить результати роботи побудованої моделі в автоматичному режимі протягом доби, оцінено результати отримані результати, зазначено особливості ринку що були виявлені підчас дослідження.

У четвертому розділі наведений фінансово-економічний аналіз програмного продукту.

ПОСТАНОВКА ЗАДАЧІ

1. Розробити систему для безпосереднього доступу до даних торгової біржі, накопити історичні данні для попереднього аналізу.
2. Обрати відповідну торгівельну пару для подальшої розробки та сформувати критерій успішної роботи системи.
3. Сформувати правила, що відтворюють торгівельні операції на біржі та протестувати отриману систему в реальному часі.

РОЗДІЛ 1 АНАЛІЗ ЕФЕКТУАЦІЙНОЇ КОНЦЕПЦІЇ ТА МЕТОДИ ЇЇ ВПРОВАДЖЕННЯ

За останні роки, підхід до логіки прийняття рішень на основі ефектуаційної (effectuation) концепції стає дедалі популярнішим. Уперше, зазначена концепція знайшла своє детальне відображення у працях С. Сарасваті, зокрема в 2001 році він запропонував нове, альтернативне поняття для традиційної логіки - каузаль (causation) поведінки підприємця [1]. Необхідність введення нової концепції було спричинено значним звуженням світу, в результаті широкого застосування нових засобів зв'язку, і виявленням значних відхилень від раціональної поведінки підприємців, також в рамках концепції враховуються значною мірою можливості змови гравців ринку, емоційна складова підприємця.

1.1 Поняття ефектуаційної концепції

У наступних працях С. Сарасваті[2][3] визначав запропоновану концепцію через її основні принципи:

- Принцип ресурсноорієнтовних дій. Замість дій спрямованих на досягнення поставлених цілей, особа, що приймає рішення, починає діяти, спираючись на початкові ресурси, які не завжди сприяють поставленій меті. Тобто ресурси формують початкову мету.
- Принцип допустимих втрат: менше ризику – мінімум витрат. Базується на ідеї формування плану в умовах концептуальної невизначеності спираючись на ризики у найгіршому з сценаріїв (підприємець самостійно визначає максимально допустимий рівень втрат ресурсів, грошових, часу і на основі цих критеріїв приймає рішення) без витрат часу на побудову моделі оцінки невизначеності. Здається, що це значно знижує темпи розвитку підприємства,

однак, слід зазначити, що швидкість прийняття рішень за умов відсутності завчасно розробленої стратегії роблять підхід надзвичайно гнучким та надають можливість користуватися будь-якими новими можливостями.

- Принцип стратегічних альянсів. Підприємець базує свою стратегію не на основі конкурентного аналізу, а на основі партнерських відносин у будь-якій сфері. Це дає можливість створювати нові ринки, або розширювати старі (але в нашому випадку цей принцип закладає необхідність враховувати змови на ринку).
- Принцип контролю, а не передбачення. Підприємець базується на ресурсах які вже є в наявності, тобто в нього відсутня потреба в передбаченні ситуації, адже, де-факто, він вже готовий до будь-якого розвитку в межах свого ресурсу.

Більшість сьогоднішніх стратегій та біржових систем відноситься до каузаційних: базуються на побудові складних моделей прогнозування, спрямовані на очікувану дохідність. Однак у випадку наявності невизначеності в середовищі прогнозування та розрахунок потенційного прибутку стає ускладненім або неможливим. Саме в такому середовищі засади звичайної концепції стають не ефективними. Най яскравіше це можна спостерігати на ринку криптовалют Цьому є декілька передумов:

- Ціна формується лише попитом та пропозицією, власне ресурс майже не має собівартості виробництва, а отже ціна на нього може коливатися в значному діапазоні
- Значна волотильність ціни, зокрема зумовлена попередім фактом
- Не значна залежність від зовнішніх факторів, зокрема валютних коливань, політичних подій

Відповідно до каузаційної концепції цінні папери з значною волатильністю є найбільш ризикованими і їх варто уникати, однак в межах ефектуаційної концепції це лише потенційна можливість при застосуванні [4].

1.2 Високочастотний трейдинг та методи аналізу якості обраного активу

Відповідно до ефуктуаційної концепції швидкість прийняття рішень є домінуючою складовою ведення операцій. Важливість швидкості для фінансових ринків не новим явищем. В 2009 році перевага швидких операцій (відкриття позиції та її закриття відбувалось менш ніж за секунду) стала очевидною – до 60% всіх операцій були зроблені в автоматичному режимі і існували долі секунди [5]. Це навіть зумовило появу нового терміну High-frequency trading (HFT) – високочастотне проведення операцій (трейдінг).

Основними засадами HFT э відкриття коротко строкових позицій з мінімальним прибутком (для максимально швидкого збуту акцій). За рахунок незначного часу утримування акцій зменшується ризи пов’язанні з знеціненням активу. Зокрема збільшується коефіцієнт Шарпа - Універсальним показником заведеним для оцінки ефективності стратегії чи інвестиційного портфелю:

$$S = \frac{E|R - R_f|}{\sigma} ,$$

де R – дохідність портфеля(активу);

R_f – дохідність альтернативного вкладу, що має низьку ступінь ризику, зокрема банківська ставка;

σ – стандартне відхилення дохідності портфелю(активу).

Однак, в умовах значної дисперсії активу коефіцієнт зменшується. Тобто при значній кількості гравців на ринку з стратегіями побудованими на HFT збільшується дисперсія активів і збільшується ризикованість операцій(а на малих проміжках часу дохідність активів стає випадковою і вже не відповідає

загальним біржовим правилам) Саме тому з 2012 року популярність HFT поступово знижується.

Для оцінки залежносте між біржовими параметрами біло використано коефіцієнти рангової кореляції Спірмена та Пірсона [6] (для виявлення можливих зав'язків та відкидання не суттєвих параметрів).

$$r_{x,y} = \frac{\sum_{i=1}^{m}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{m}(x_i - \bar{x})^2 \sum_{i=1}^{m}(y_i - \bar{y})^2}} \ ,$$

де $r_{x,y}$ – коефіцієнт кореляції Пірсона;

$\{x_i\}_{i=1}^{m}$, $\{y_i\}_{i=1}^{m}$ – ряди що перевіряються на лінійну залежність на проміжку значень [1, m] з вибірковим середнім $\bar{x}$, $\bar{y}$ відповідно.

Коефіцієнт набуває значень [-1, 1] значення -1 та 1 відповідають сильній лінійній залежності оберненого та прямого характеру відповідно, 0 – характеризує некорельованість параметрів, тобто відсутність лінійної залежності

Коефіцієнт кореляції Пірсона нажаль є нестійким до викидів, які підчас проведення дослідження були виявленні Рис. 1.3.1 – яскраво демонструє один з викидів. Також за відсутності корельованості за Пірсоном не можна стверджувати незалежність параметрів. Частково проблеми кореляції Пірсона вирішує кореляція за Спірменом. Він дозволяє оцінити існування залежності між змінними, що може бути виражено монотоною функцією [6].

Основними перевагами використання кореляції Спірмена є ранговість коефіцієнтів, інваріантність по відношенню до будь-якого монотонного перетворення і більша чутливість до лінійних залежностей, що яскраво проілюстровано на Рис. 1.1 та Рис. 1.2.

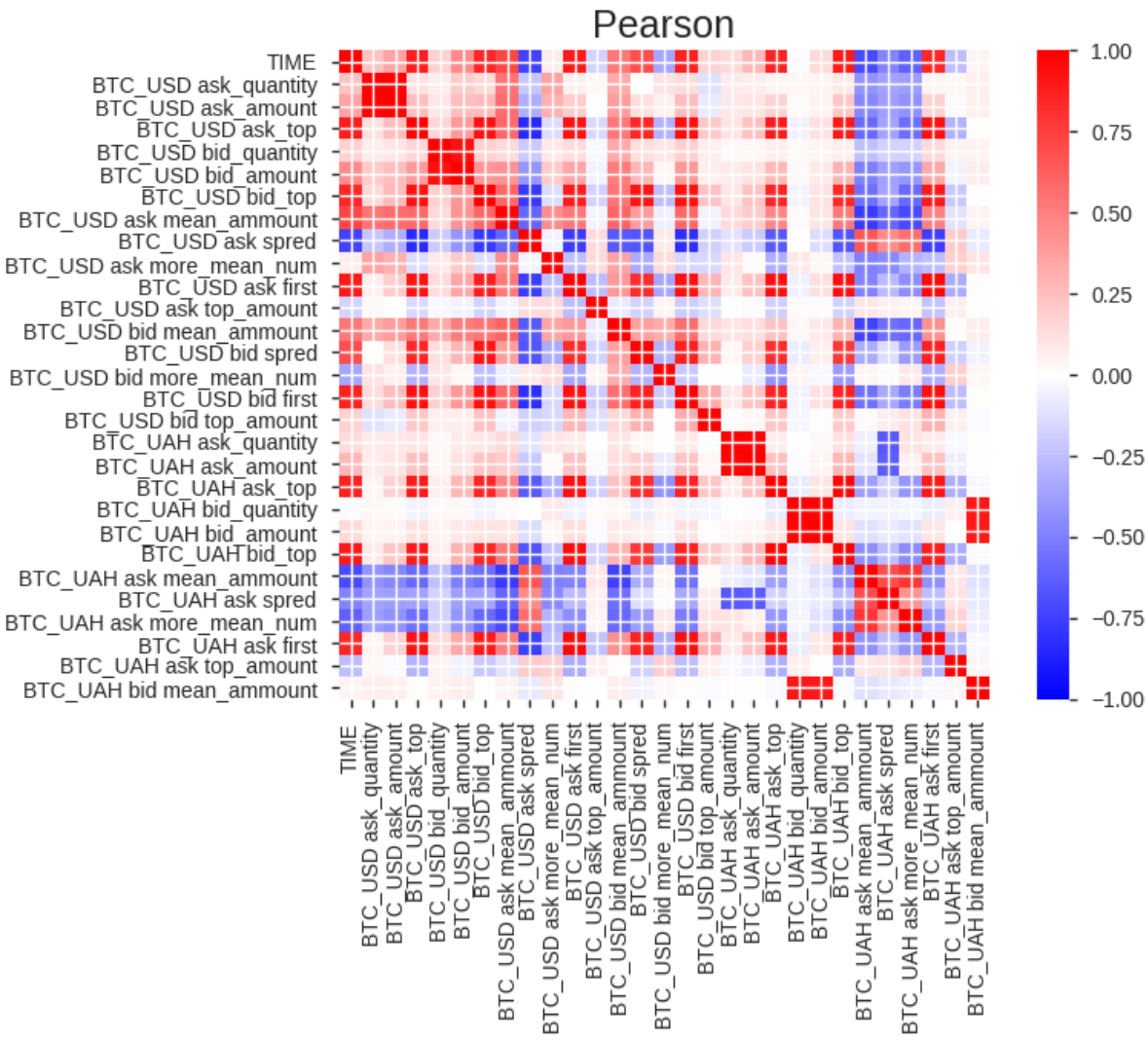


Рисунок 1.1 – Кореляційна матриця Пірсона для перших 2 валютних пар параметрів

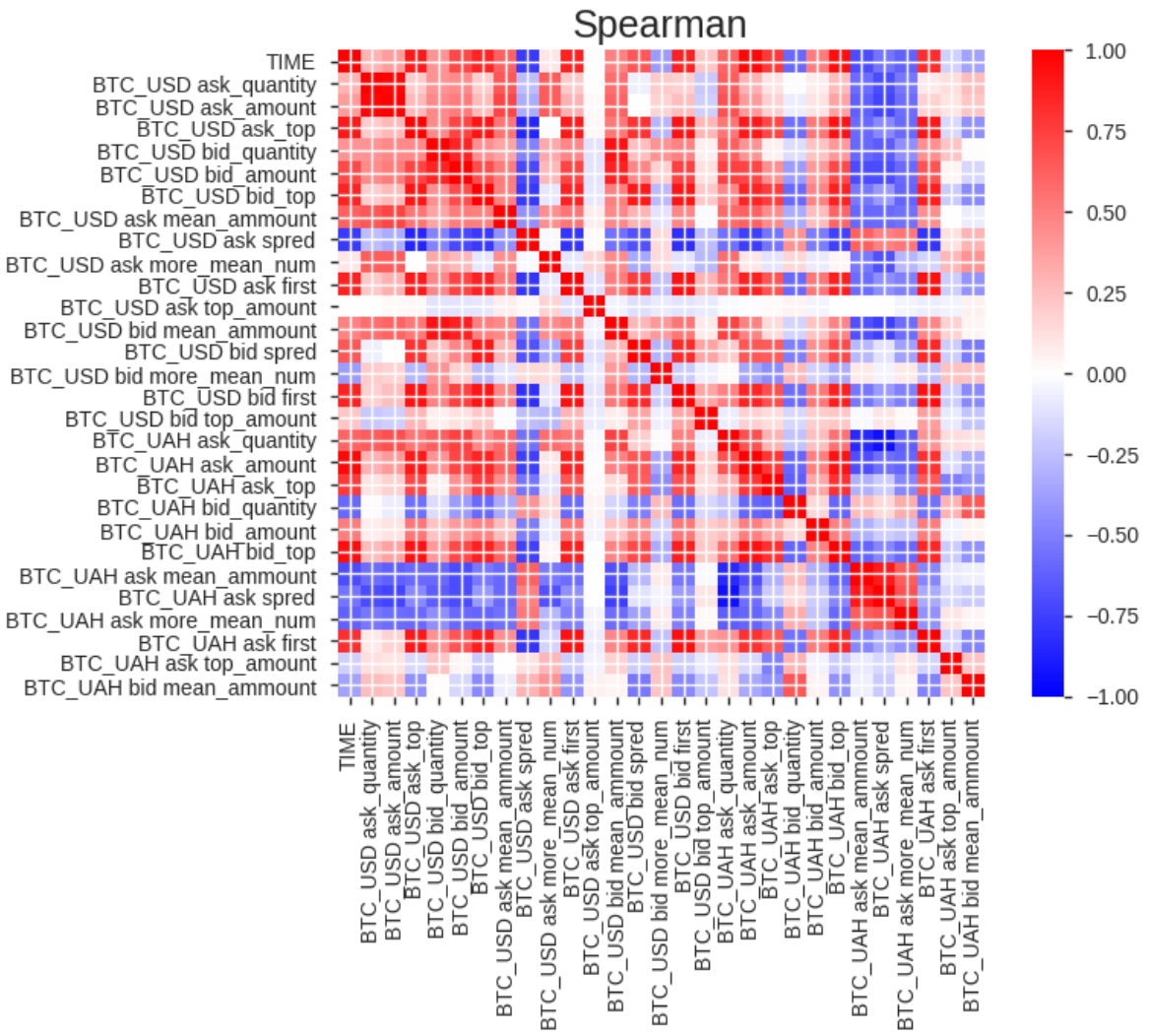


Рисунок 1.2 – Кореляційна матриця Спірмана для перших 2 валютних пар параметрів

1.3 Поліноміальна регресія в якості попередньої обробки даних

Для проведення мінімізації чи максимізації складної ступінчатої функції накшталь курсу валют зазвичай використовують регресійні моделі. Такі моделі також використовувати для створення прогнозів для стаціонарних процесів або процесів з повільною зміною режимів з малим рівнем щуму. Нижче наведено результати згладжування часового ряду за 1 годинний інтервал поліномом 45 степені(Рис 1.3).

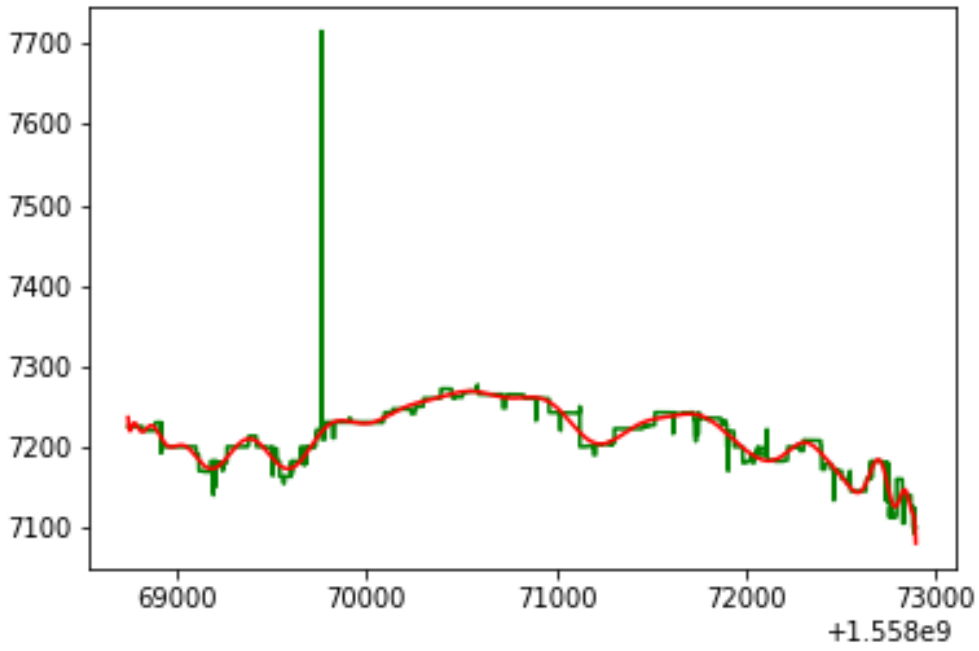


Рисунок 1.3 – Згладжування отримані при впровадженні поліноміальної регресійної моделі

У роботі використовувалась поліноміальна регресія. ЇЇ можна описати наступним чином (для ідеального випадку, без наявності шуму та викидів):

$$y_i = \sum_{k=0}^{N} a_k x_i^k,$$

де x_i - незалежна змінна що спостерігається (регресор);

y_i- змінна що спостерігається (частково) і є залежною;

a_k- коефіцієнти моделі;

i – індикатор виміру (оскільки в нашому випадку модель відтворює неперервну функцію на основі високочастотних вимірювань);

N – максимальна степінь поліному моделі.

Аналогічно до лінійної регресії, при побудові моделі в якості критерію оптимальності моделі було обрано сума середньоквадратичного відхилення:

$$S = \sum_{i=0}^{M} (\widehat{y_i} - y_i)^2 \to min,$$

де S – критерій якості моделі;

$\widehat{y_i}$ – теоретичне оптимальне значення (реальні виміри);

y_i – змодельовані значення;

M – розмір експериментальних даних.

У метода є декілька особливостей. По-перше, для коректної побудови моделі необхідно принаймні 40-50 вимірювань на кожну ступінь регресора (при введенні декількох регресорів використовують аналогічне правило – як і для збільшені степені полінома, Рис 1.4).

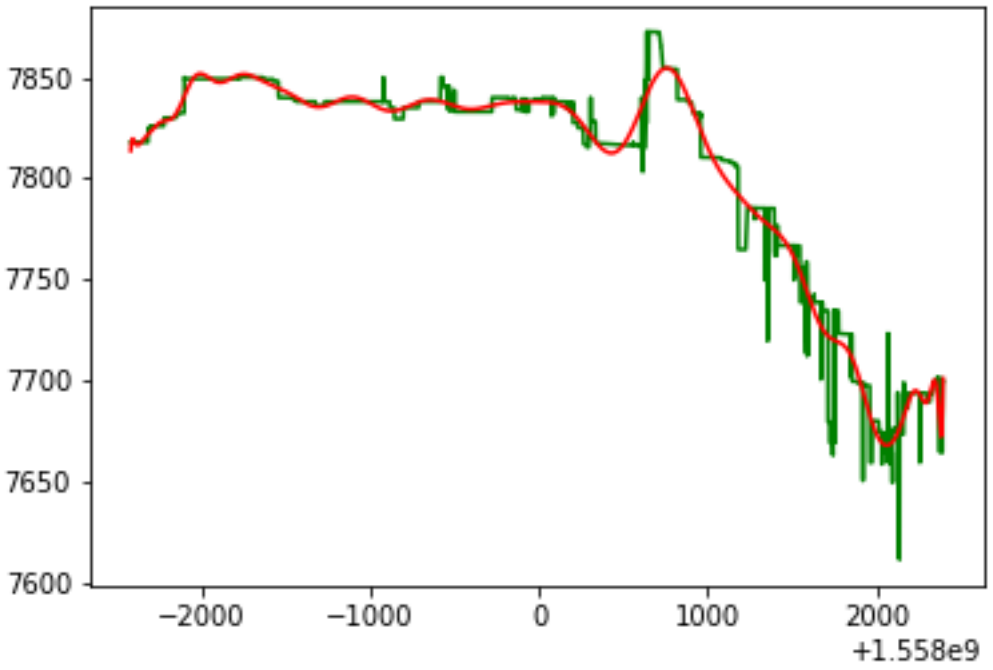


Рисунок 1.4 – Ілюстрація негативності ефекту “згладжування” у моделі з 0.05% відхиленням

По-друге, деякі короткочасні зміни не відображаються у побудованій моделі (Рис. 1.4 середнє відхилення досліджуваної величини 0.05% однак помітні значні втрати на проміжках [0,1000] та [1200,2200]). Також наявні обмеження які накладаються з технічного боку для кожного дослідження індивідуально, пов’язані з розмірністю вимірюваних параметрів. Зокрема для часових рядів застосовується центрування:

$$\hat{x}_i = x_i - \frac{1}{m}\sum_{i=1}^{m} x_i \,,$$

де $\hat{x}_i$ – елементи відцентрованої множини;

x_i – регресор, введення якого призводить до переповнення типу даних;

m – розмірність розглянутого періоду.

1.4 Формалізація задачі біржової торгівлі для високочастотного трейдингу

Більшість операцій що проводяться на ринку можна визначити через усього 2 їх види: купівля, продаж. Регулюючи, певним чином, моменти їх проведення трейдер отримує прибуток. А отже задачу отримання прибутку можна звести до задачі максимізації, що в деякому наближенні можна розглянути як різницю двох поліномів (Рис 1.5).

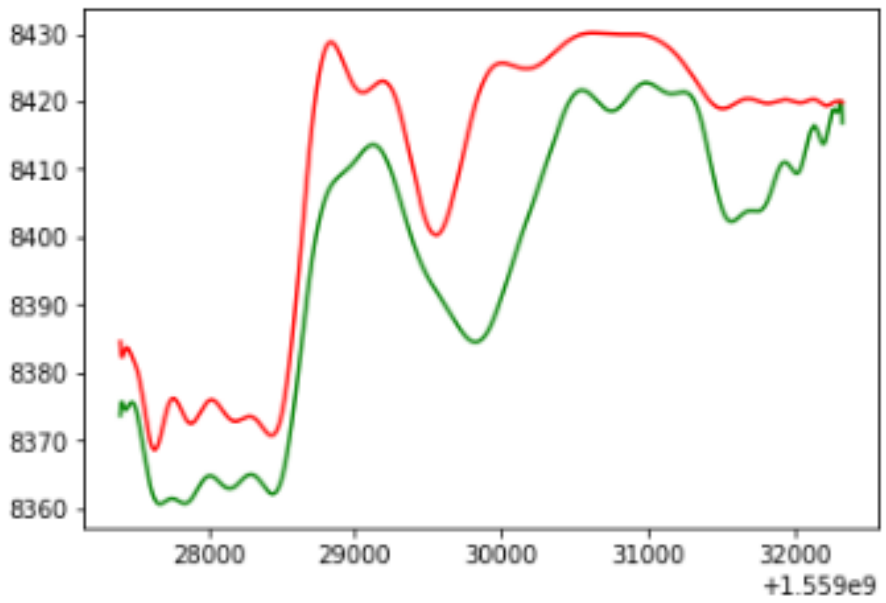


Рисунок 1.5 – Апроксимовані дані, придатні до розв’язання формальної задачі (червоним ціна купівлі, зеленим ціна продажу)

Задача в явному вигляді формується наступним чином:

$$F(T_1, T_2) = \frac{P_1(T_1)}{P_2(T_2)}(1 - commision)^2 \rightarrow max,$$

де $P_1(T_1)$ – ціна продажу ресурс в момент часу T_1;

$P_2(T_2)$ – ціна купівлі ресурс в момент часу T_2;

$commision$ – комісія за операцію біржі.

Функція $F(T_1, T_2)$ відповідає частці капіталу після проведення пари операцій: купівлі в час T_1 та продажу в T_2. Тобто приймає значення $(0, \infty)$, причому $(0,1)$ операцію можна вважати неуспішною. Для періоду на Рис. 1.5 функція $F(T_1, T_2)$ представлена на Рис. 1.6, де темно коричневим – успішні операції, а від коричневого до синього операції що призводять до втрат.

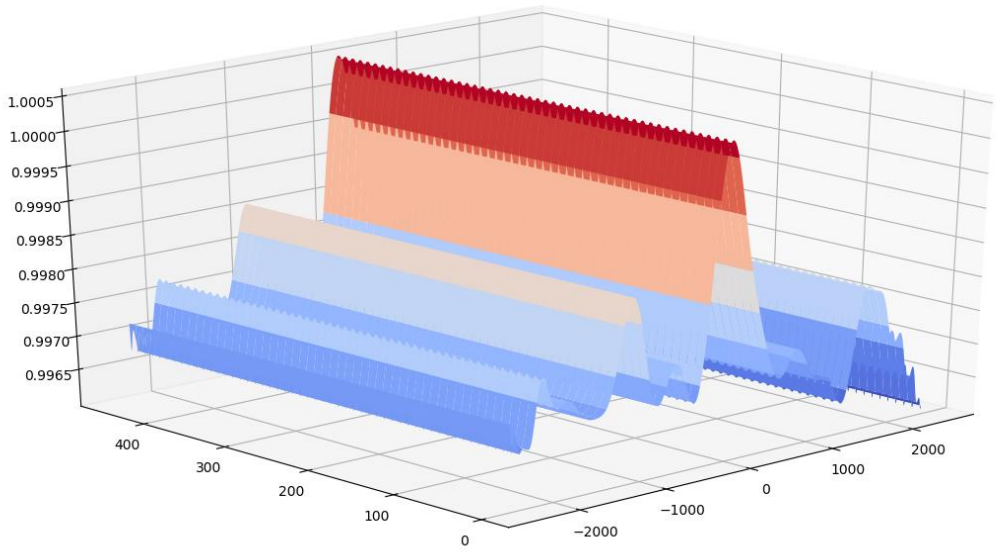


Рисунок 1.6 – Результати проведеного моделювання функції виграшу трейдера для проміжку часу, що відповідає Рис. 1.5 (від 0 до 400 – час між операціями в с, -2000 до 2000 – відцентрований час проведення першої операції)

Пошук оптимальних моментів проведення операцій можна звести до пошуку нулів функції, чи локальних мінімумів. Більшість алгоритмів ґрунтується

на модифікованих методах гріентного спуску. Градієнтний спуск – ітераційний алгоритм оптимізації першого порядку для знаходження локального мінімумам функції. Оскільки після проведення поліноміального згладжування моделі функція виграшу при торгівлі активом може бути задана в явному вигляді (причому диференційована) тому данні методи можуть бути застосовані для пошуку оптимальної торгівельної стратегії в першому наближені.

1.5 Метод LGBM

LGBM (Light gradient boosting method – облегшений метод градієнтного збільшення) об’єднання технік, що застосовується в машинному навчанні для розвязання проблем класифікації(співвідношення об’єктів до визначених сукупностей) та регресії [7]. ЇЇ відносять до слабких прогнозуючих моделей (на відміну від інших методів теоретична досяжна точність методу менша). На відміну від інших методів машинного навчання – процес навчання в LGBM не може бути розпаралелений та використовувати потужності відео карти і використовує лише можливості процесора, адже в процесі роботи використовуються ітеративні алгоритми навчання і дерева прийняття рішень(в моєму випадку), побудова яких лише ітераційна.

Метод базується на використанні технології boosting (технологія далеко відійшла від першого значення цього слова – збільшення). В її основу покладено побудова незалежних окремих моделей одного чи різного типу де процес навчання наступної ґрунтується на помилках попередньо навченої моделі [8].

Як і більшість методів метод базується на зменшенні середньо квадратичної похибки (зокрема в моєму дослідженні, як найбільш прийнятний до часового ряду):

$$\frac{1}{M}\sum_{i=0}^{M}(\widehat{y_i} - y_i)^2 \to min$$

В загальному вигляді процес навчання можна записати наступним чином(для методу GBM):

$\{(x_i, y_i)\}_{i=1}^{n}$ – навчальна вибірка, не повинна перетинатись з тренувальною вибіркою (не обов'язкова умова, але таким чином зменшується ризи перенавчання моделі – запам'ятовування моделлю конкретних значень, а не відтворення залежностей [9])

$L\big(y, F(x)\big)$ – функція витрат (loss function – вказує наскільки модель відрізняється від реальних значень на тренувальній вибірці, може використовуватись середньо квадратична похибка або інша метрика відхилень, в залежності від поставленої задачі)

1. Впровадження моделі зі сталими значеннями:

$$F_0(x) = arg_\gamma min \sum_{i=1}^{n} L(y_i, \gamma)$$

2. Для кожного кроку навчання(M – попередньо задана кількість кроків, $m = \overrightarrow{1, M}$):

a. Обраховуємо псевдо-залишкову функцію:

$$r_{im} = \left[\frac{\partial L\big(y_i, F(x_i)\big)}{\partial F(x_i)}\right]_{F(x)=F_{m-1}(x)}, \text{для } i = 1, \dots, n$$

b. Навчання основної моделі h_m на основі вибірки $\{(x_i, r_i)\}_{i=1}^{n}$

c. Шукаємо коефіцієнт γ_{im} шляхом вирішення наступної оптимізаційної задачі:

$$\gamma_{im} = arg_{\gamma} min \sum_{i=1}^{n} L\big(y_i, F_{m-1}(x_i) + \gamma h_m(x_i)\big)$$

d. Оновлюємо модель:

$$F_m(x) = F_{m-1}(x) + \gamma_m h_m(x_i)$$

3. Шукана функція $F_M(x)$

В якості моделі для тренування використовували дерева прийняття рішень, що в спрощеному вигляді можна записати наступним чином [10]:

$$h_m(x) = \sum_{j=1}^{J_m} b_{jm} I_{R_{jm}}(x),$$

де: J_m – кількість листків;

R_{jm} – простір потрапляння x, що визначається кожним листком;

$I_{R_{jm}}(x)$ – індикатор потрапляння x в R_{jm}.

Справжньої популярності метод набув після 2008 року, з появою платформи “Kaggle” (платформа для змагань в обробці, аналізі та передбаченні великих даних) Поступово LGBM став символом перемоги – у більшості змагань (80%) проведених за 2012 рік всі призові місця використовували цю технологію (на платформі відзначають 10% кращих) Тобто майже у всіх задачах: від регресії до розпізнавання образів найкращі 10% аналітиків знайшли застосування LGBM

і вона виправдала себе. Згодом популярність такого підходу пішла на спад, але станом на 2018 рік кожна друга робота використовує або саму технологію, або спрощену її версію.

Основною причиною його популярності і відмінністю від інших алгоритмів побудованих на деревах рішень є вертикальне (leaf-wise – точний переклад не надто логічний для української термінології: методологія нарощування листків і вузлів дерева, що орієнована на ефективну побудову усього дерева) нарощування листків дерева (проти горизонтального в типових алгоритмах Рис.1.7, Рис.1.8)

Рисунок 1.7 – Вертикальне нарощування листків дерева(LGBM)

Рис 1.8 Горизонтальна побудова дерева(інші алгоритми GBM)

Горизонтальне нарощування дерев також називають level-wise (рівнево орієнтовне нарощування Рис 1.5.2). Основними перевагами при виборі leaf-wise технології стала більша швидкість обробки великих даних і менші потреби пам’яті на їх обробку. Однак за рахунок такої побудови дерев у алгоритму є схильність до перенавчання особливо на незначних вибірках. Емпіричним шляхом було встановлено мінімальний розмір вибірки ля успішного навчання: не менше 10000 вимірів

1.6 Висновки до розділу 1

Отже у розділі 1 розглянуто основні визначення та теоретичні викладки до розглянутої проблеми. Наведено основний напрямок дослідження.

РОЗДІЛ 2 ВПРОВАДЖЕННЯ МОДЕЛІ В РИНКОВИХ УМОВАХ

Для тестування та дослідження було обрано біржу крипто валют. Вибір був обумовлений надзвичайною волатильністю активів порівняно з ринком акцій, централізованим і , водночас, незначним впливом зовнішніх факторів(новини пов'язані з крипто валютним ринком певним чином відображаються відразу на всіх крипто активах) також наявні сезонні явища (пов'язані з можливістю емісії для будь-якого користувача, використання крипто валют для розрахунків). Ці фактори дають навіть більший простір для відпрацювання підходів ніж власне типові торгові біржі.

Серед великої кількості активів було обрано найбільш популярні: BTC, ETH , та найпопулярніші, для України, показники ефективності стратегій: UAH, USD, EUR. Відповідно було сформовано 7 валютних пар для проведення торгових операцій: BTC_USD, BTC_UAH, BTC_EUR, ETH_USD, ETH_UAH, ETH_EUR, ETH_BTC. Для кожної з пар біржа надавала доступ до наступних даних:

- ask_quantity – обсяг(кількість) всіх позицій на продаж
- ask_amount – ціна всіх позиці на продаж
- ask_top – мінімальна ціна на продаж
- bid_quantity – обсяг(кількість) всіх позицій на купівлю
- bid_amount – ціна всіх позиці на закупівлю
- bid_top – максимальна ціна на покупки

Додатково ведено наступні характеристики для кожної з валютних пар та відповідної групи позицій на продаж та купівлю:

- mean_ammount – середній об'єм перших 100 позицій(зазвичай саме на короткострокових інтервалах(2-3 хвилино) вони відграють значну роль на формування ціни.

- spred – показує ступінь розрідженості перших 100 позицій (різниця між 1 та 100 позицією – такий показник потенційно зменшить витрати часу на навчання моделі)
- more_mean_num – кількість позицій що перевищують mean_ammount на перших 100 позиціях(показник виявив корельованість до зростання та спадання ціни, а отже може простити навчання моделі вподальшому)
- first – перша позиція що перевищує mean_ammount(зазвичай виконує роль своєрідного обмежувача коливань ціни на незначних відрізках часу 1-2секунди)
- top_amount – розмір найвищої позиції на продаж/покупку, дозоляє відтворювати операції на біржі (також від його розміру залежить стабільність ціни, а від змін її рух)

Додатково введені показники були обрані на основі кореляційної матриці Спірмана та Пірсона. Таким чином отримано 113 параметрів. (Рис 2.1 та Рис 2.2 – для всіх зазначених параметрів, Рис 1.3, Рис 1.4– для перших 2 валютних пар)

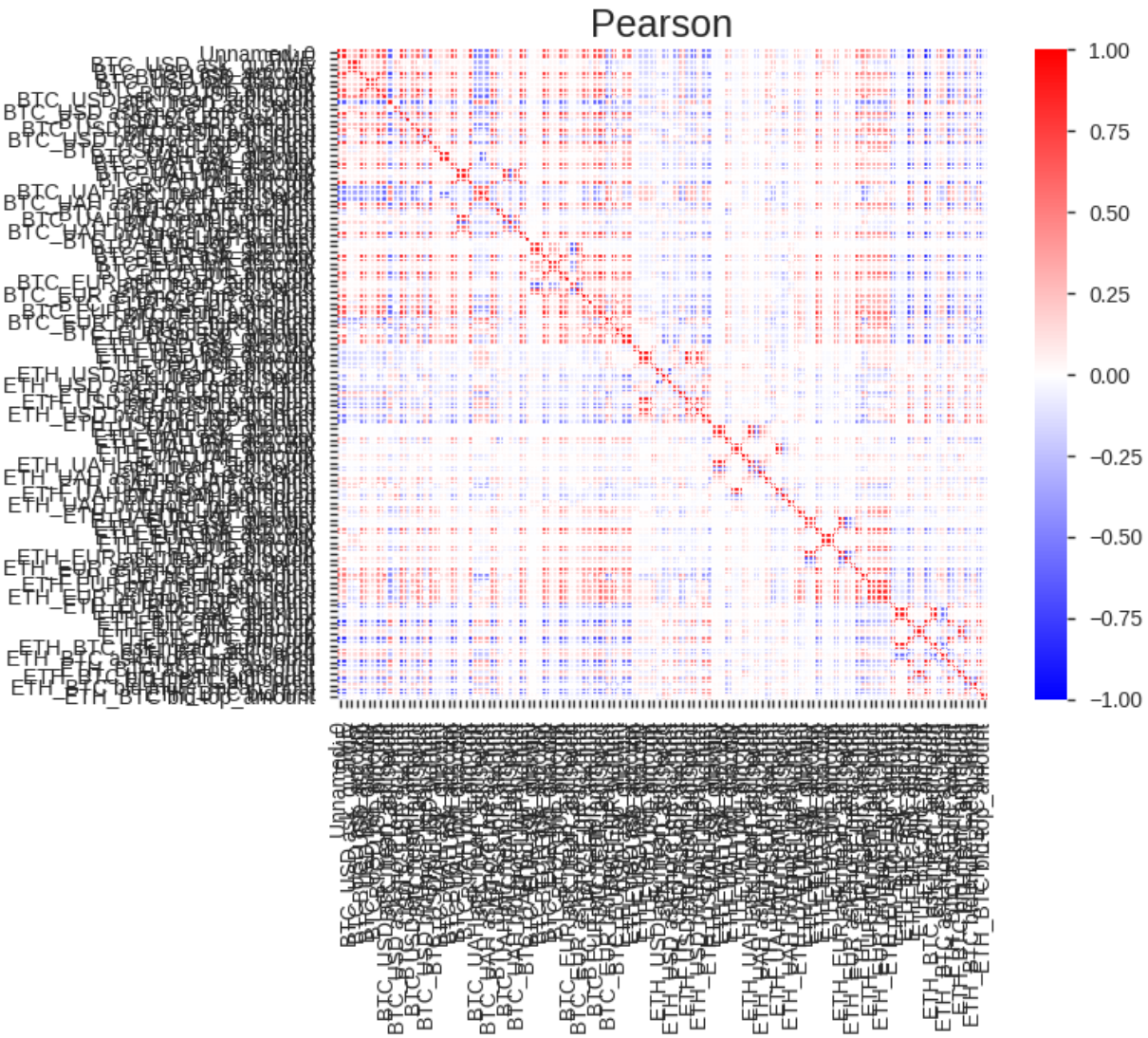


Рисунок 2.1 – Кореляційна матриця Пірсона

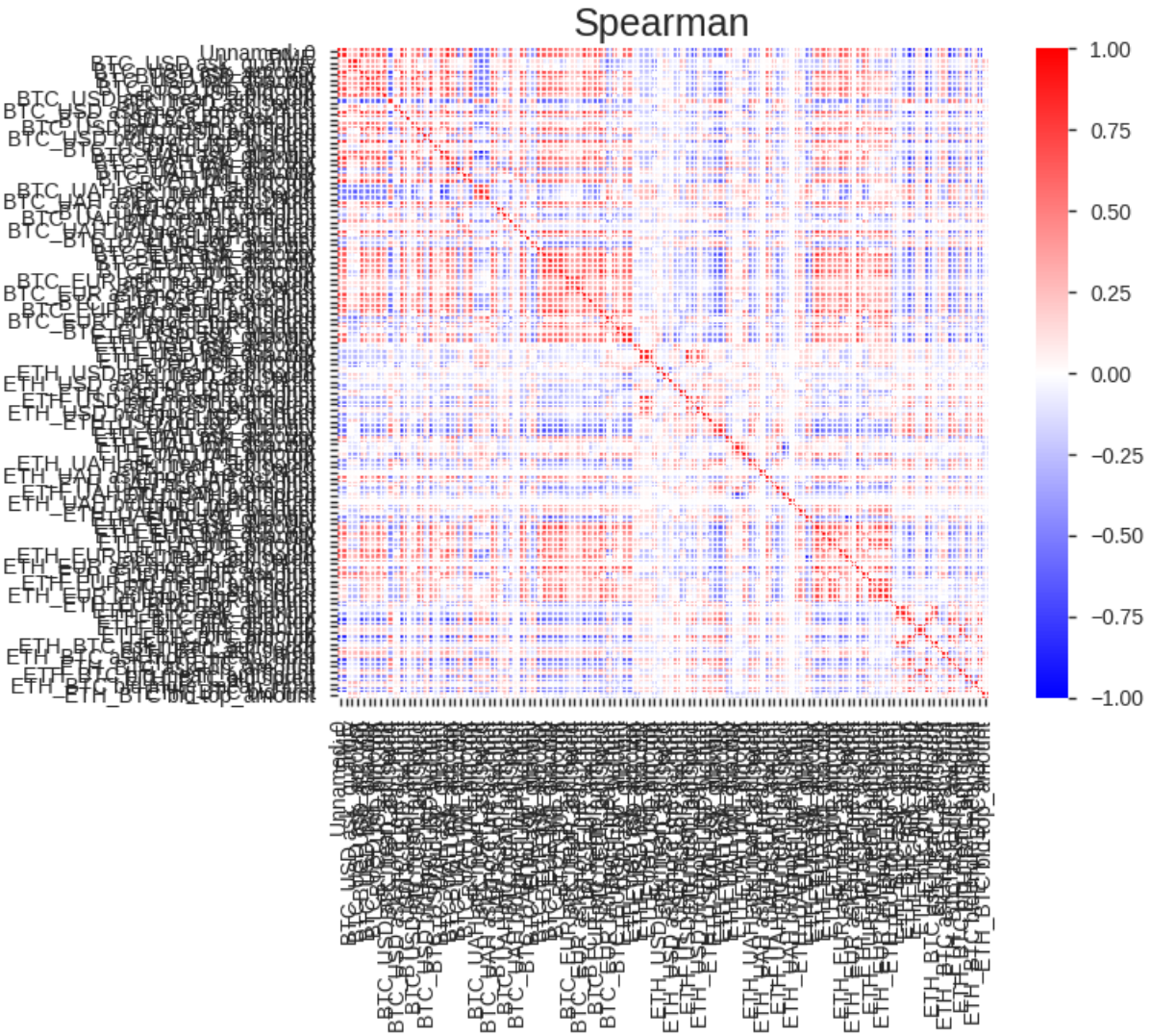


Рисунок 2.2 – Кореляційна матриця Спірмена

Також було виявлено сильно картельовані параметри, що в перспективі можуть бути виключені з моделі, у випадку перенавчання або задля зменшення часу навчання (Рис 2.3):

- BTC_UAH ask_first is highly correlated with BTC_UAH ask_top (ρ = 0.90912) Rejected
- BTC_UAH ask_amount is highly correlated with BTC_UAH ask_quantity (ρ = 0.98289) Rejected
- BTC_UAH ask_quantity is highly skewed (γ1 = -29.66) Skewed
- BTC_UAH ask_top is highly correlated with BTC_USD ask_first (ρ = 0.94028) Rejected
- BTC_UAH bid mean_ammount is highly correlated with BTC_UAH bid_amount (ρ = 0.90294) Rejected
- BTC_UAH bid_amount is highly correlated with BTC_UAH bid_quantity (ρ = 0.98042) Rejected
- BTC_UAH bid_quantity is highly skewed (γ1 = -24.172) Skewed
- BTC_UAH bid_top is highly correlated with BTC_UAH ask_top (ρ = 0.975) Rejected
- BTC_USD ask_first is highly correlated with BTC_USD ask_top (ρ = 0.92952) Rejected
- BTC_USD ask_amount is highly correlated with BTC_USD ask_quantity (ρ = 0.98756) Rejected
- BTC_USD bid_first is highly correlated with BTC_USD ask_first (ρ = 0.93533) Rejected
- BTC_USD bid spred is highly correlated with BTC_USD bid_top (ρ = 0.91976) Rejected
- BTC_USD bid_amount is highly correlated with BTC_USD bid_quantity (ρ = 0.93499) Rejected
- BTC_USD bid_top is highly correlated with BTC_USD ask_top (ρ = 0.92746) Rejected

Рисунок 2.3 – Параметри що сильно корелюють

2.1 Вибір валюної пари

Для проведення торгівельних операцій було обрану одну валютну пару, базуючись на наступних критеріях:

- найбільша волатильність
- одна з валют повинна бути стабільною для її накопичення (в якості успішності роботи системи виступає саме її накопичення)
- значна кількість незначних операцій користувачів

Для цього було розглянуто 55 хвилинні відрізки з максимальною частотою:

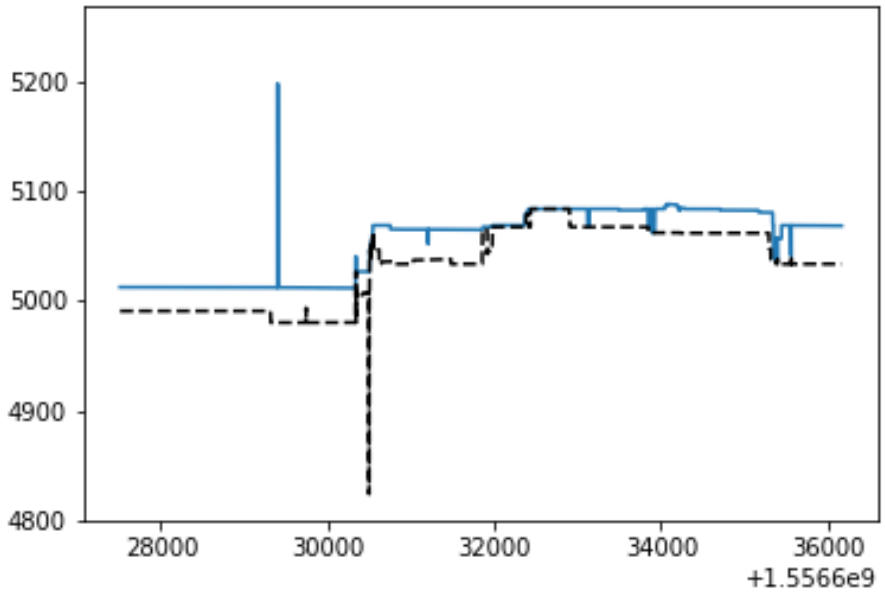


Рисунок 2.4 – Коливання BTC по відношенню до EUR за 55 хвилинний термін (чорним – ціна продажу активу, синім – купівлі)

Відповідно до Рис. 2.4 на досліджуваному періоді пари BTC EUR наявна незначна кількість мікро коливань (на межі мінімально дозволеної кількості). Курс відносно стабільний на великих проміжках часу (в середньому зміни

відбуваються раз на 700 секунд). Таким чином BTC EUR не найкраща валютна пара для проведення високочастотного трейдингу.

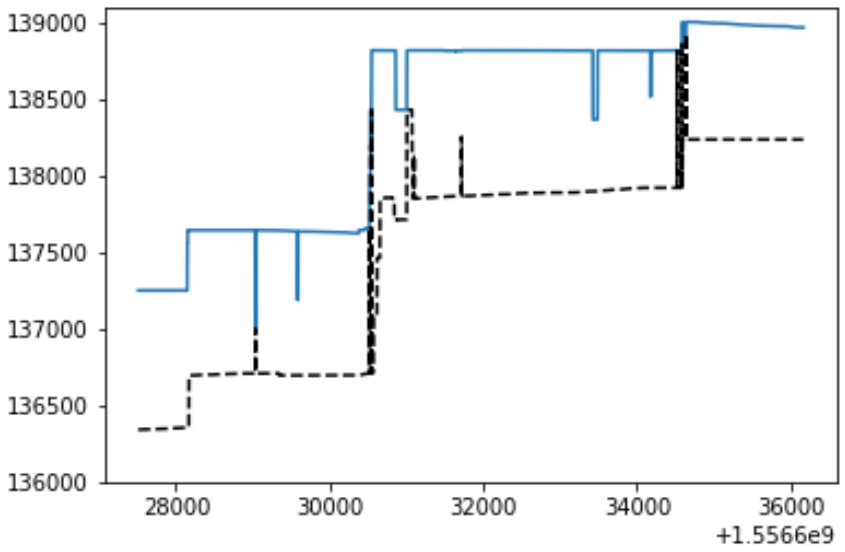


Рисунок 2.5 – Коливання BTC по відношенню до UAH за 55 хвилинний термін (чорним – ціна продажу активу, синім – купівлі)

Відповідно до Рис 2.5 для валютної пари BTC UAH характерна незначна коротко строкова активність також наявні значні коливання спричинені незначною кількістю відкритих позицій (а отже курс не стабільний , наявні значні коливання, пов’язані с коректуванням курсу самою біржою, щоб унеможливити крос валютну торгівлю, наприклад EUR UAH)

Аналогічно було проведено аналіз всіх 7 валютних пар. Найкращі характеристики за обраними показниками були виявлені в парі BTC USD (Рис 2.6).

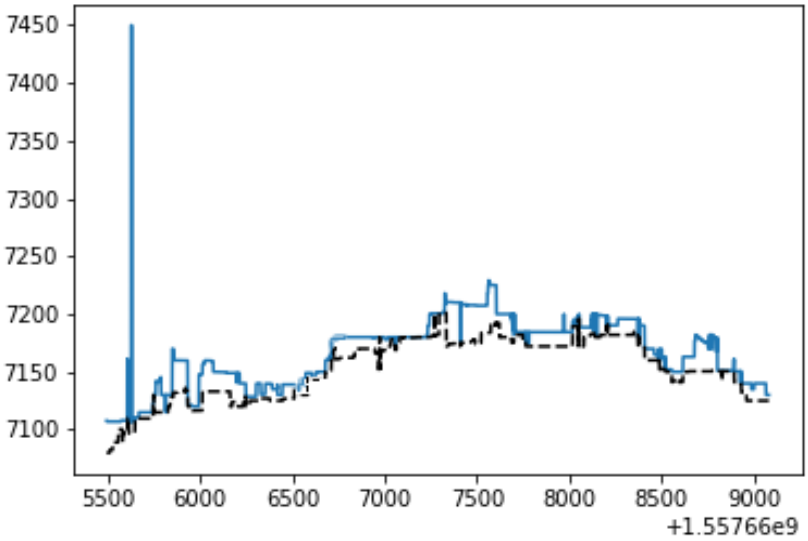


Рис 2.6 Коливання BTC по відношенню до USD за 55 хвилинний термін (чорним – ціна продажу активу, синім – купівлі)

Валютна пара BTC USD мала найбільшу кількість змін за досліджуваний час (порядку 95% були наявні зміни в ціні покупки чи продажу, що перевищують розмір комісії, див Рис 2.7)

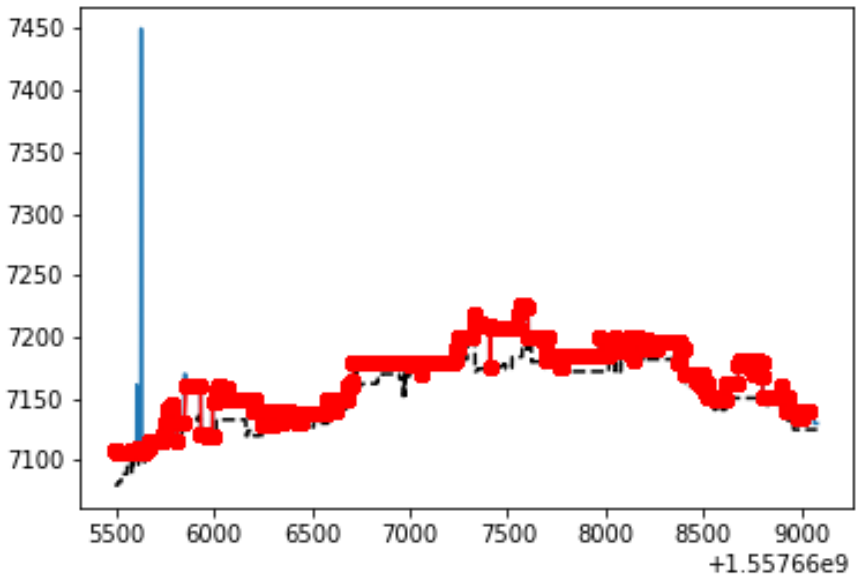


Рис 2.7. Коливання BTC USD, за досліджуваний період (чорним – ціна продажу активу, синім – купівлі)

Таким чином торгівельні операції будуть проводитись лише на парі BTC USD інші валютні пари будуть використанні як допоміжні показники, адже наявність корегуючи дій (Рис 2.6, Рис 2.7) безумовно свідчать про взаємний вплив торгівельних пар.

2.2 Передобробка даних та їх розмічення

Для проведення розмічення даних до подальшого навчання моделей потрібно провести згладжування моделі. Для цього використовуються механізми лінійної регресії. В випадку слабких коливальних процесів зазвичай застосовують поліноміальну регресію (Рис 2.8)

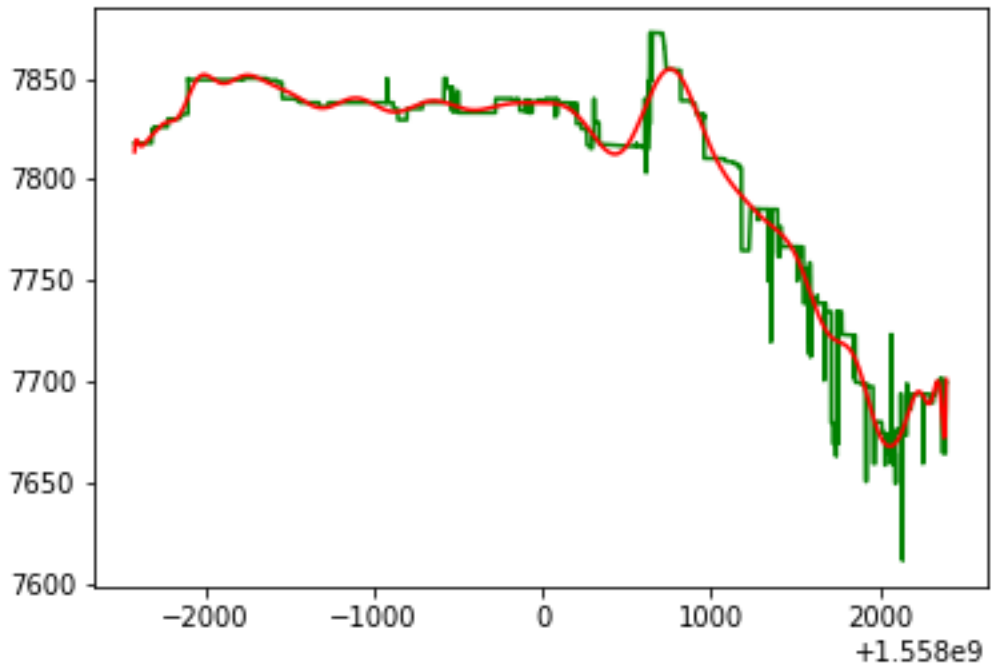


Рисунок 2.8 – Залагоджування реальних даних(зеленим) поліномами 45 степені(червоним)

Однак в результаті проведення регресії шукані відрізки значною мірою зменшились. Зокрема за досліджуваний період на Рис 2.2.1 кількість шуканих моментів часу зменшилась з 1.5% до 0.09%, а отже зменшився розмір тренувальних динних. Тому було прийнято рішення використовувати поліноміальну модель(Рис 2.2.2) лише для пошуку вірогідного розміщення

шуканих відрізків, а потім уточнювати їх місце знаходження числовими методами на реальних даних:

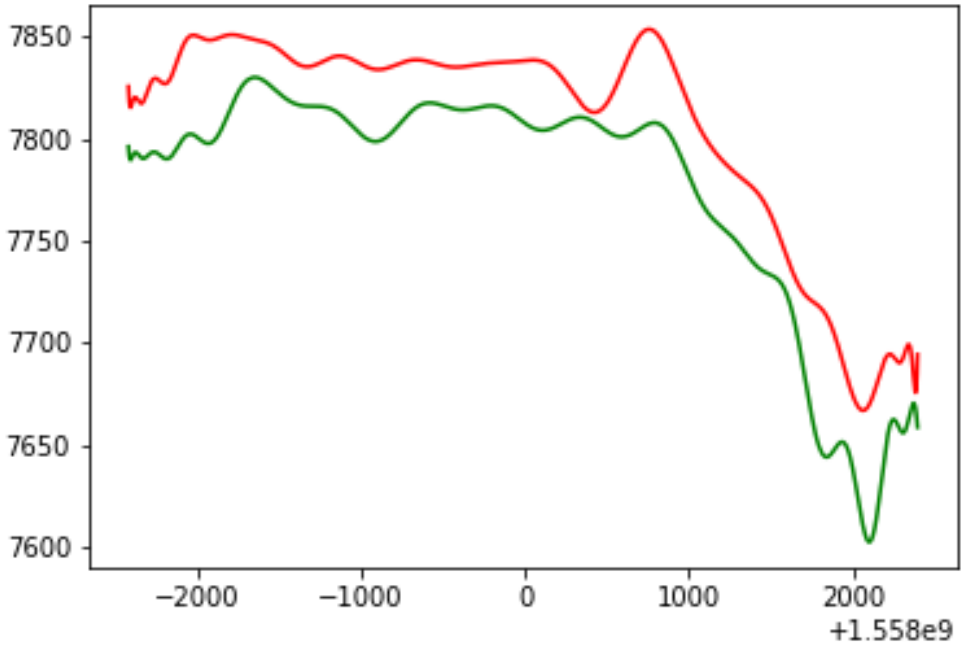


Рисунок 2.9 – Дані після поліноміальної регресії 45 ступеня (червоним ціна продажу, ціна купівлі зеленим)

Зокрема на Рис 2.9 для подальшого дослідження будо обрано проміжки з найбільшим наближенням кривих купівлі і продажу. Однак дана модель не враховую викиди, шо можуть бути потенційно найприбутковішими(Рис 2.10).

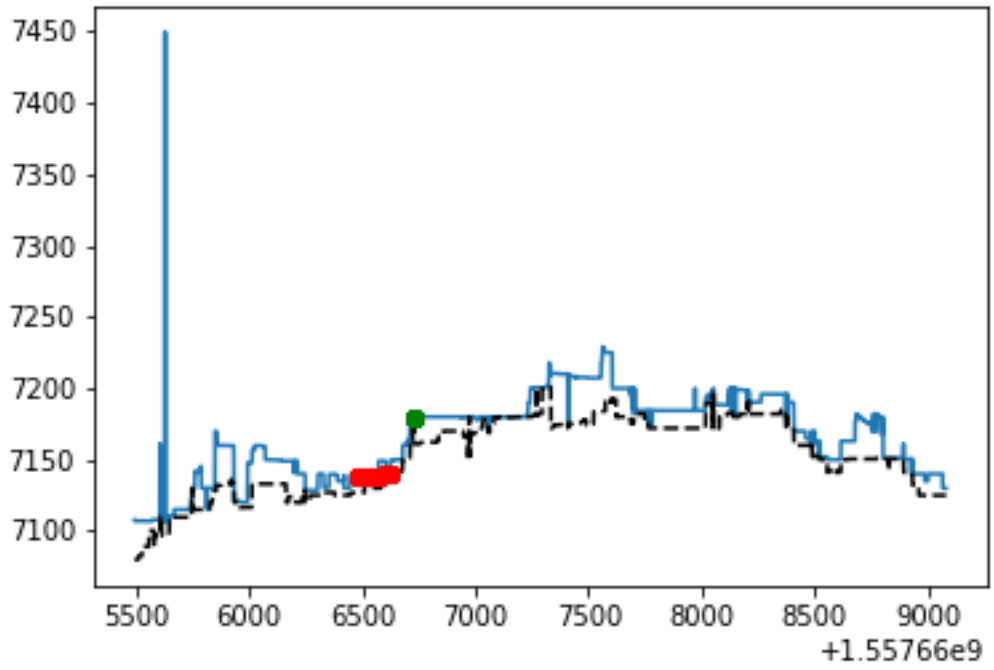


Рисунок 2.10 – Ілюстрація наявних викидів(на проміжку 5500 5700). Червоним зазначено найвигіднішу купівлю на проміжку і її продаж, за умови 2.5хвилиного утримування активу

2.3 Застосування методу lgbm

Для методу LGBM через його особливості надзвичайно важливим правильна побудова моделі, адаптована під випадок його застосування, розмірність даних, потрібна точність, структура даних, їх тип. В загальному, для повного налаштування моделі можна застосовувати більш ніж 100 параметрів [11], [12]. Їх розподіляють на декілька груп.

До першої групи відносять контрольні параметри (Control Parameters), що відповідають за базові властивості побудови дерев:

- max_depth – характеризує максимальну глибину дерев. Відповідно для збільшення точності моделі потрібно максимізувати цей параметр, водночас це перший параметр, що потрібно зменшити у разі перенавчання(тобто якщо помилка на тренувальній вибірці зменшується, а на тестовій збільшується – це першу насідки перенавчання, що є означає потреби зменшити глибину моделі)
- min_data_in_leaf – мінімальна кількість записів, шо може мати листок (за замовчанням 20 записів). Також регулювання параметра допомагає перебороти перенавчання.
- feature_fraction – дозволяє визначити вказану частку параметрів випадковим чином(необхідний параметр для побудови моделі випадкового лісу(random forest). Наприклад 0.7 означає, що на кожній ітерації 70% параметрів генеруються випадковим чином.
- bagging_fraction – специфікатор що відповідає за розподілення даних для кожної ітерації. Зазвичай використовується для збільшення швидкості тренування моделі. Може призвести до її погіршення.
- early_stopping_round – дозволяє автоматизувати навчання. Алгоритм зупиниться якщо одна з метрик або проходження по тестовій виборці перестане давати результати. На вихід буде повернуть попередню ітерацію моделі.

- Lambda –регуляризація даних. Зазвичай належить проміжку від 0 до1.
- min_gain_to_split – параметр описує мінімальний рівень для розбиття на гілки дерева. Зазвичай використовують для зменшення кількості гілок в дереві (при значних витратах пам’яті, або перенавчані)
- max_cat_group – за замовчанням приймає значення 64. При значній кількості категорій при їх автоматичному розбитті швидко може набувати перенавчення. Їх об’єднання дозволяє уникнути цієї проблеми (особливо в випадку більше 100 категорій на 10000 вибірку)

Далі йдуть параметри що визначають поведінку моделі(Core Parameters):

- Task – визначає базовий стан моделі: тренування чи прогнозування
- Application – визначає тип моделей для навчання, відповідно до завдання, що задається змінююся метрики ефективності, засоби формування дерев. Розрізняють наступні типи:

1. Regression – адаптовані для проведення регресії даних (лінійної)
2. Binary – биранная класифікація
3. Multiclass – множинна класифікація (використовувалась в роботі: адже де-факто є всього 3 стани: очікування, купівля, продаж. Звісно можна звести і до 2 класів але тоді середовище прийняття рішень значно скорочується)

- Boosting – параметр що визначає тип комбінування моделей, за замовчання gdbt (traditional Gradient Boosting Decision Tree – попередньо описаний в першому розділі):

1. Gbdt – градієнтний бустінг (GBM)
2. Rf – (random forest – випадковий ліс) комбінування моделей засноване на побудові паралельно значної кількості дерев рішень з випадковою складової (дає значний результат, ефективна методика вибору та трактуванню відповідей базуючись на результатах навчання сформована недосконало і

значною мірою залежить від дослідника, що в умовах побудови адаптивного алгоритму недомустимо)

3. Dart – (Dropouts meet Multiple Additive Regression Trees) мультиплікативні адаптивні дерева (для регресії) з втратами

4. Goss –(Gradient-based One-Side Sampling) одностороній прохід по виборці заснований на градієнтному методі

- num_boost_round – кількість проходів по тренувальним даним для бустингу. Зазвичай обирають число більше 100 або дослідженням число, після якого точність моделі перестає зростати.

- learning_rate – визначає для кожного дерева кінцевий результат. GBM метод базується на початкових значеннях(які обираються випадково) і змінюються після кожного дерева (ітерації). Цей параметр регулює ширину кроку при оцінці параметрів дерев(аналогічно до кроку при числових методах пошуку мінімуму, одна в n мірній площині і для не надто гарно визначених функцій) Зазвичай при досліджені обирають: 0.1,0.001, 0.003. Однак на практиці особисто зустрічав приклади, що працюють оптимально саме для конкретних значень (наприклад: 0.00083)

- num_leaves – кількість рівнів в усьому дереві, за замовчанням 31.

- Device – параметр відповідає що ресурси, що використовуються для навчання:

1. Cpu – навчання на ресурсах процесора

2. Gpu – з використанням відео карти (не всі типи відео карт підтримуються і не всіх виробників)

Параметри метрики(Metric parameter). Він всього один у групі, однак саме коректне його визначення дозволяє правильно навчити модель:

- Metric – визначає критерій оцінки ефективності моделі. Нижче наведенні найпопулярніші з них [13]:

1. Mae - mean absolute error (середнє абсолютне відхилення)

$$MAE = \frac{\sum_{i=1}^{n} |\widehat{y_i} - y_i|}{n}$$

2. Mse – mean squared error (середньоквадратичне відхилення)

$$MSE = \frac{1}{n}\sum_{i=0}^{n} (\widehat{y_i} - y_i)^2$$

3. binary_logloss – (loss for binary classification) функція втрат для бінарного класифікатора(наявно декілька типів з різною сутінню строгості до помилки)

4. multi_logloss – (loss for multi classification) аналогічно для багато класової класифікації.

Інші не менш важливі але не об’єднанні в групи параметри:

- max_bin– визначає найбільший розмір змінної в яку вона може вміститися (дозволяє контролювати розміри потрібної пам’яті і опосередковано швидкість навчання).
- categorical_feature – визначає колонки в яких знаходяться категорії безпосередньо перерахуванням починаючи з 0 .
- ignore_column – аналогічно попередній характеристиці дозволяє виключити стовбець з опрацювання повністю.
- save_binary – при значних обмеженнях на розмір оперативної пам’яті дозволяє зберегти файл з даними до двійкового формату що збільшить в подальшому швидкість обробки і зчитування.

2.4 Висновки до розділу 2

Отже, в розділі 2 було наведено основні параметри моделі та застосування обраних методів. Проведено аналіз їх впливу на модель.

РОЗДІЛ 3 АНАЛІЗ ОТРИМАНОЇ СИСТЕИ ПРИЙНЯТТЯ РІШЕНЬ

Як основну мову програмування було обрано Python, оскільки вона працює на всіх відомих операційних системах та має велике різномаїття бібліотек присвячених роботі с даними. Для розробки програмного продукту були використані такі додаткові модулі мови Python: бібліотека обробки даних pandas, модуль роботи з матриця numpy та бібліотеки павязні з машиниим навчанням. Також наявна адаптована бібліотека для застосування LGBM.

Програмний продукт складається із допоміжної бібліотеки та декількох jupyter notebook. Кожен jupyter notebook є виокремленою задачою з розділа постановки задач. Завдяки використанню Python разом з платформою jupyter ці застосунки працюють на персональних комп'ютерах під управлінням різних операційних систем: MacOS, Windows, Linux. Завдяки цьому розроблений продукт може бути перенесений разом з моделлю на окремий сервер, однак вже без засобів візуалізації – jupyter, що дозволить покращити швидкодію, однак призведе до ускладнення моніторингу результатів роботи моделі.

3.1 Аналіз вхідних данних

В якості джерела даних для перевірки обраного підходу було обрано біржу крипто валют "EXMO" критеріями для вибору саме цієї платформи стали:

- Міжнародний статус біржі, офіс знаходиться в Лондоні.
- Наявність повноцінної підтримки користувачів з України (офіційне представництво в Києві.
- Наявність необхідних валютних пар.

Після оформлення необхідної документації було біржою було надано доступ до даних в реальному часі з максимальною кількістю запитів на їх

оновлення 180 на хвилину. Після впровадження доступу було виявленні помилки пов’язані з розірванням з’єднання та тимчасовою відмовою видачі даних на боці біржі. Дані проблеми були успішно вирішенні, однак максимальна частота отримання даних була зменшена до 133 на хвилину.

Досягти однакової частоти дискретизації в наданих біржою умовах не представлялось можливим без втрати частини даних, тому було вирішено звернутись до без прогнозованих трейдингових моделей моделей.

Для зменшення об кількості даних для обробки , а отже і зменшення об’єму необхідної кількості пам’яті було обрано лише 6 валютних пар з 700 можливих, а актуальні торгівельні пропозиції було зведено з 600 (по 6 парам відповідно) до 155 (розділ 2.1), що полегшило модель та дало змогу додати механізм пам’яті до моделі без її перенавчання. Використання торгівельних пропозицій для прийняття рішення не використовується в більшості торгівельних алгоритмів через значне ускладнення моделі з введенням нових параметрів не категоріальної природи. Однак в випадку “не пргонозованих” моделей це важливе джерело даних, що за умови відсутності рівномірної дискретизації значно покращило модель, а попередня обробка дозволила це зробити без значного ускладнення.

3.2 Порівняльний аналіз роботи моделі

Передумови накладені на вхідні данні (попередній розділ) вплинули на метод обраний для розв’язання поставленої задачі. Для її розв’язання розглядались моделі побудовані на наступних методах:

- Метод найближчих сусідів.
- Лінійний SVM метод.
- RBF SVM.
- Метод побудований на Гаусівському процесі.
- Дерево прийняття рішень.

- Випадковий ліс (комбінація дерев прийняття рішень).
- Нейроні мережі.
- ADa boost.
- Баесівські мережі.
- QDA.

Аналізувати кожний алгоритм окремо у 155 мірному просторі важко, оскільки результати можна оцінити лише за скалярними метриками [13] і втрачається інформація про модель в не розглянутих при навчанні точках.

Для аналізу та порівняння моделей було застосовано стандартні вибірки даних в 2 мірному просторі. Отримання неуступні результати (Рис 3.1).

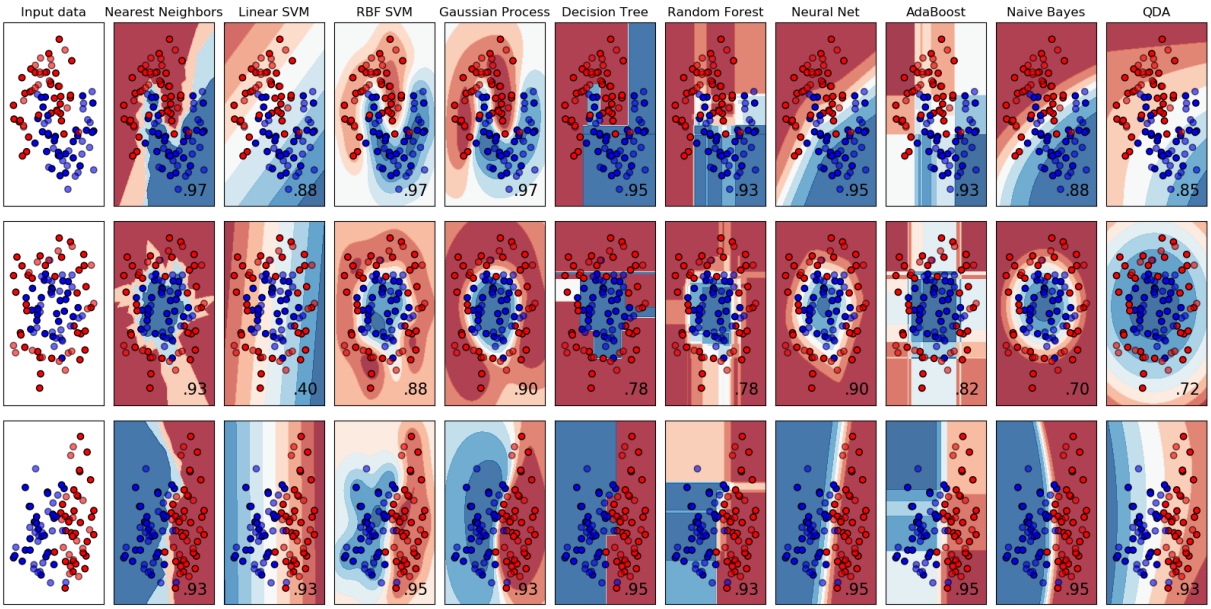


Рисунок 3.1 – Порівняння методів для вибору моделі для класифікації, на прикладі класифікації синіх та червоних точок в 2 мірному просторі

Відповідно до Рис. 3.1 найгірші результати, з позиції довгострокового прогнозування має метод AdaBoost, оскыльки має білі області, що у випадкі

класифікації дасть велику групу точок, що не будуть віднесені до жодного з класів і відразу стануть похибкою моделі(причому частина тренувальних точок потрапила в ці області, що зовсім погано).

Методи Decidion tree та Random forist базуються на алгоритмі що дозволяє розділяти данні в 2 мірному просторі лише горизонтальними та вертикальними лініями (точніше Random forist базується на Decidion tree). Однак в методі Random forist спостерігається перенавчання: велика кількість невеликих областей з високою імовірністю (з однією тренувальною точкою) та значна кількість областей з відсутньою характеристикою.

Лінійний SVM метод показав себе повністю не пристосованим до поставленої задачі.

Методи найближчих сусідів, RBF SVM, гаусівський процес показали непогані результати однак потребували значно більшу кількість обчислювальних ресурсів і в по загальному відхиленню не надто відрізнялись від дерев прийняття рішень.

На основі поданого аналізу методів було прийняте рішення використовувати методи що ґрунтуються на деревах прийняття рішень. Метод Випадкового лісу де дерева будуються паралельним чином було відкинуто, а отже залишижться алгоритми з послідовною побудовою дерев, а саме XGBoost та LGBM (рзіниця між якими описана в розділі 2).

3.3 Аналіз роботи моделі

Аналіз роботи моделі проводивля на добовому інтервалі роботи в режимі 75 хвилин роботи 15 хвилин навчання. Далі наведено результати попереднього формування моделі на 1 годинному інтервалі. Формалізована задача класифікаціїї має настцний вигляд на першому інтервалі:

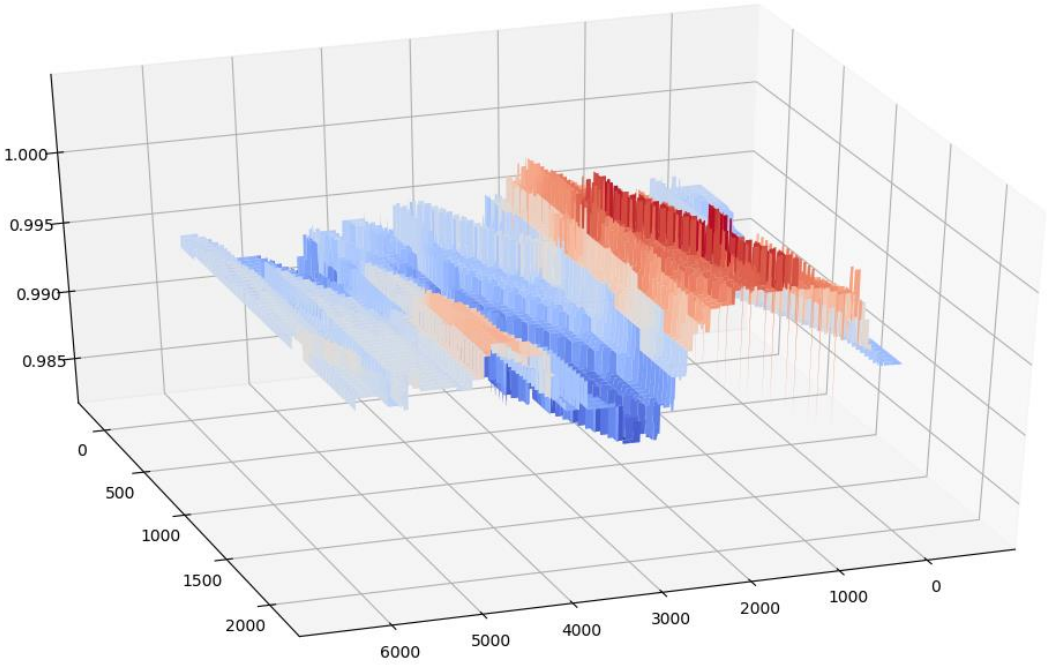


Рисунок 3.2 – Формалізована задача класифікації

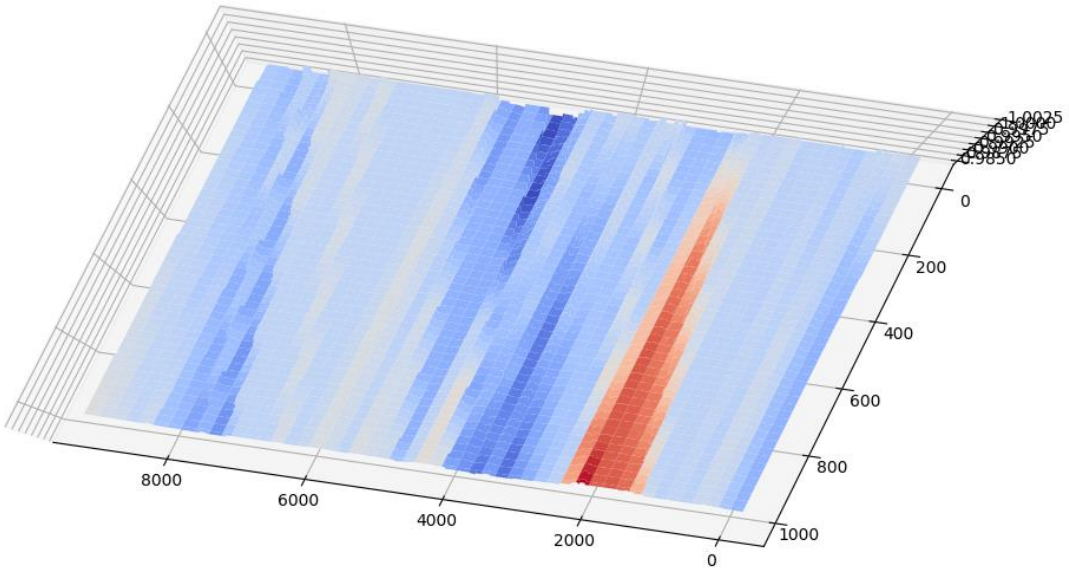


Рисунок 3.3 – Формалізована задача класифікації

Відповідно до Рис. 3.2 та Рис 3.3 чітко видно проміжки купівлі продажу(купівля здійснена на приблизно на 900 секунді (2000 вимір) при продажі через 450 секунд, 1000 вимірів, принесе прибуток в розмірі 0.25% від суми операції). Для проведення класифікації було побудована модель “без пам’яті”. Однак не було отримано жаданих результатів:

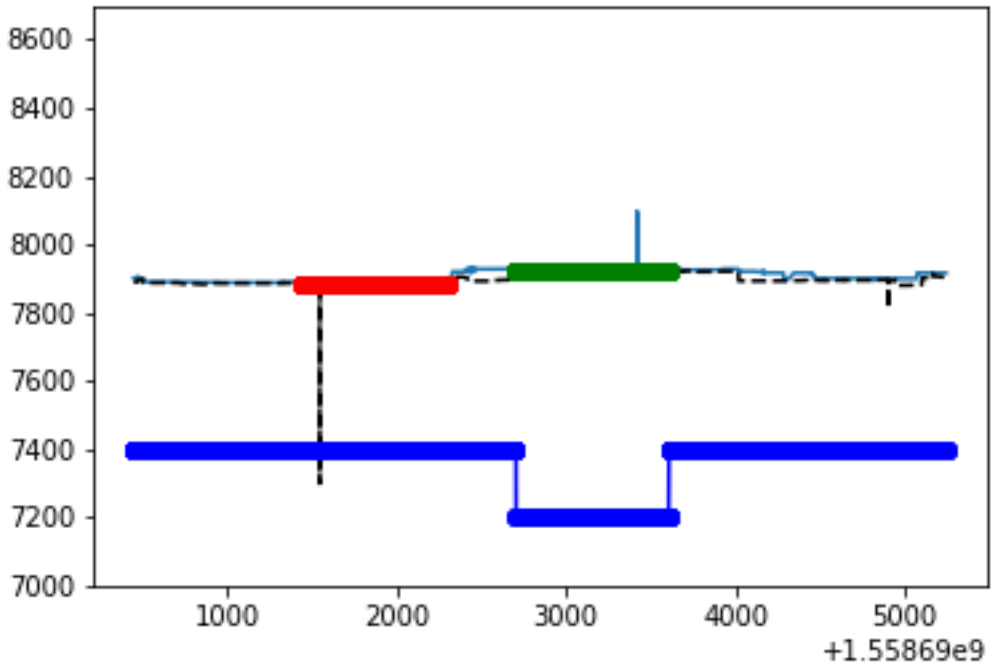


Рисунок 3.4 – Робота моделі без пам’яті, синім – результати класифікації(один з класів точок, що відповідає закупівлі(червоним) відсутній)

Тому було прийнято рішення про введення механізму пам’яті на 2 кроки, таким чином кількість параметрів збільшилась до 270 , однак при розмірі навчальної вибірки 10000 (75 хвилиний термін роботи) перенавчання було уникненню.

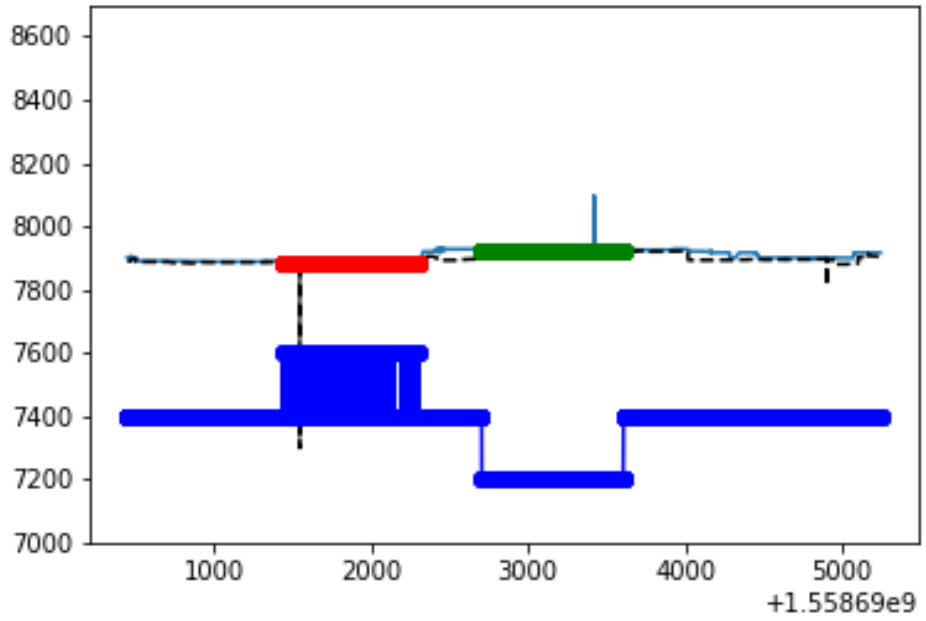


Рисунок 3.5 – Результати класифікація на навчальній вибірці моделі з пам’яттю синім (чітко видно 3 класи точок)

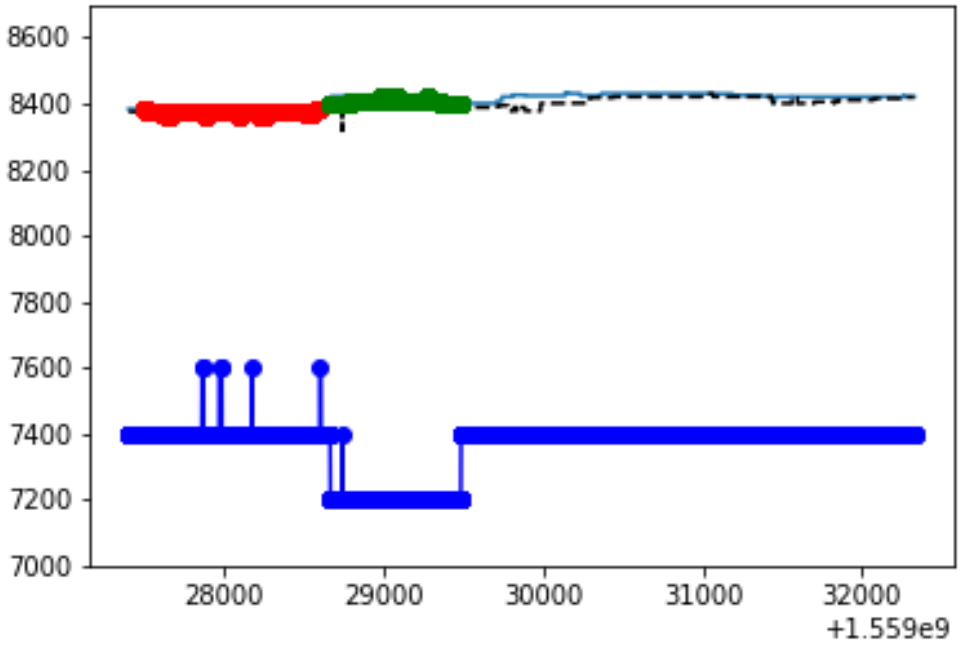


Рисунок 3.6 – Робота алгоритму з пам’яттю на відрізці часу, на якому не було проведено попереднього навчання

Відповідно до Рис 3.5 та Рис 3.6 можна оцінити складність розпізнавання кожного класу точок: очікувано клас вчасної купівлі за короткий час до стрибка ціни є більш складною задачею ніж власне стрибок курсу. Однак LGBM знайшов приховані правила, що не базуються на часі та не прив'язані до значення курсу, а лише до його зміни. На Рис 3.6 значна кількість точок, що відповідають купівлі не була класифікована, однак не було допущено більш грубої помилки: змішування класу продажу та купівлі, а отже з точки зору бізнесу модель відпрацювала коректно, адже фінанси були примноженні і момент не було пропущено (достатньо 1 точки в червоній зоні для отримання прибутку).

Підчас тестування алгоритму в різний час доби було виявленно закономірні колвання активності торгів на біржі повязані з локальністю гравців. Зокрема у нічнй час, з 2 до 4 години ночі, активність не значна і ціна зазвичай нижча ніж о 10 ранку, на який приходиться пук активності. Ця особливість відобразилась на побудованій моделі після 1 добової роботи:

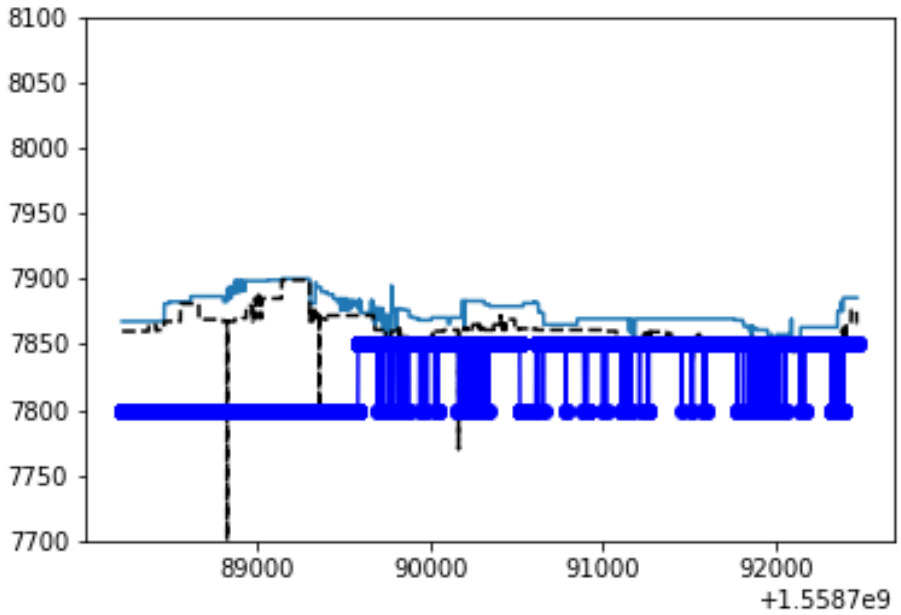


Рисунок 3.7 – Нічна робота алгоритму – масова рекомендація по закупівлі від дерева рішень

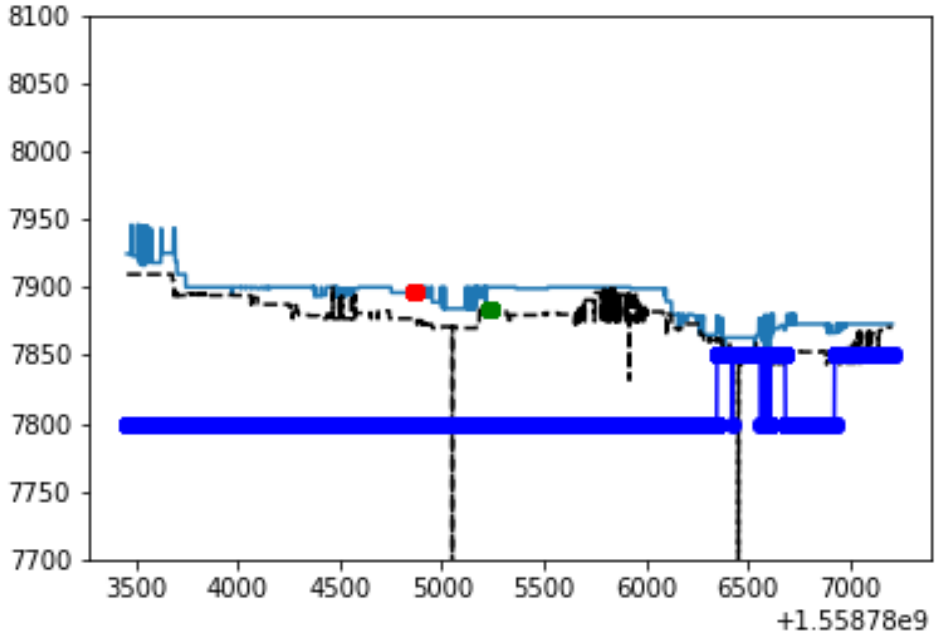


Рисунок 3.8 – Перехід біржі в “нічний режим”, початок рекомендації по масової закупівлі від моделі

Хоча робота моделі з довгим триманням активів (протягом ночі) не розглядалась при розміченні даних (данні для навчання було розміченню з прорахунком на 10 хвилинне тримання) виявлена тенденція до закупівлі свідчить про аналогічний набір правил для довгострокової стратегії та атракторної поведінки ринку криптовалют (оскільки ціна формується тільки за рахунок пропозицій та попиту обмеженого кола людей то виявленні закономірності в їх сукупній поведінці на короткострокових операціях можуть бути застосовані на довгострокових, принаймні така закономірність виявлена)

Важлива деталь показана на Рис.3.8: незначні коливання, час яких незначний, класифікатором не розпізнаються як прибуткові, адже час проходження операції може бути більшим за прибуткове вікно.

3.4 Висновок до розділу 3

Отже в розділі 3 розглянуто роботу моделі: її навчання та поведінка на досліджуваному періоді.

РОЗДІЛ 4 ФУНКЦІОНАЛЬНО-ВАРТІСНИЙ АНАЛІЗ ПРОГРАМНОГО ПРОДУКТУ

У даному розділі проводиться оцінка основних характеристик програмного продукту. Даний продукт розроблений на мові програмування Python в якості самостійної програмного продукту. Програма в першу чергу призначено для підтримки користувача у прийнятті рішень на ринку цінних паперів та адаптований для ринку крипто валют “EXMO”, може бути як частиною експертної системи, так і окремим сервісом.

Нижче наведено аналіз різних варіантів реалізації модулю з метою вибору оптимального, з огляду при цьому як на економічні фактори, так і на характеристики продукту, що впливають на продуктивність роботи і на його сумісність з апаратним забезпеченням. Для цього було використано апарат функціонально-вартісного аналізу.

Функціонально-вартісний аналіз (ФВА) – це технологія, яка дозволяє оцінити реальну вартість продукту або послуги незалежно від організаційної структури компанії. Як прямі, так і побічні витрати розподіляються по продуктам та послугам у залежності від потрібних на кожному етапі виробництва обсягів ресурсів. Виконані на цих етапах дії у контексті метода ФВА називаються функціями.

Мета ФВА полягає у забезпеченні правильного розподілу ресурсів, виділених на виробництво продукції або надання послуг, на прямі та непрямі витрати. У даному випадку – аналізу функцій програмного продукту й виявлення усіх витрат на реалізацію цих функцій.

Фактично цей метод працює за таким алгоритмом:

- визначається послідовність функцій, необхідних для виробництва продукту. Спочатку – всі можливі, потім вони розподіляються по двом групам: ті, що впливають на вартість продукту і ті, що не впливають.
- для кожної функції визначаються повні річні витрати й кількість робочих годин.
- для кожної функції на основі оцінок попереднього пункту визначається кількісна характеристика джерел витрат.
- після того, як для кожної функції будуть визначені їх джерела витрат, проводиться кінцевий розрахунок витрат на виробництво продукту.

4.1 Постановка задачі техніко-економічного аналізу

У роботі застосовується метод ФВА для проведення техніко-економічного аналізу розробки.

Відповідно цьому варто обирати і систему показників якості програмного продукту.

Технічні вимоги до продукту наступні:

- програмний продукт повинен функціонувати на сучасних обчислювальній машині, з параметрами не нижчини за наступні: частота процесора 1.7Ггц, оперативна пам'ять 4Гб, мсце на жосткому диску 225Гб та доступом до мережі інтеренет 75Мб/с;
- забезпечувати високу швидкістьприйняття рішень (менше 300мс на рішення);
- забезпечувати максимальну відсутність похибок другого роду, які призводять до втрат капіталу;
- забезпечувати зручність і простоту встановлення на будь-яку апаратну систему та поєднання з іншими програмними системами(Linux, Windows та їх аналоги);

- передбачати мінімальні витрати на впровадження програмного продукту.

4.1.1 Обґрунтування функцій програмного продукту

Головна функція F_0 – розробка програмного продукту,дозволяэ розпізнавати прибуткові передумови до коливань та приймати рішення про закупівлю чи продаж активу. Виходячи з конкретної мети, можна виділити наступні основні функції ПП:

F_1 – вибір мови програмування;

F_2 – вибір оптимальної бібліотеки обробки даних;

F_3 – постачальник економічних даних.

Кожна з основних функцій може мати декілька варіантів реалізації.

Функція F_1:

а) мова програмування C++;

б) мова програмування Python;

Функція F_2:

а) Zipline;

б) Pandas.

Функція F_3:

а) EXMO;

б) Yahoo finance.

4.1.2 Варіанти реалізації основних функцій

Варіанти реалізації основних функцій наведені у морфологічній карті системи (рис. 4.1). На основі цієї карти побудовано позитивно-негативну матрицю варіантів основних функцій (таблиця 4.1).

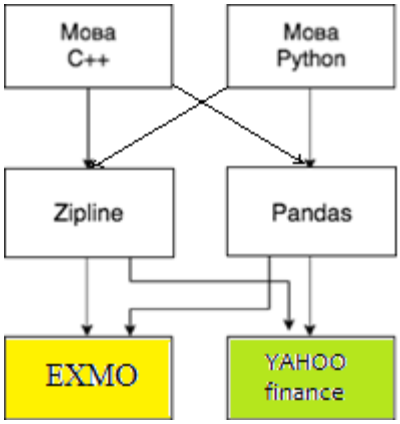


Рисунок 4.1 – Морфологічна карта

Морфологічна карта відображує всі можливі комбінації варіантів реалізації функцій, які складають повну множину варіантів ПП.

Таблиця 4.1 – Позитивно-негативна матриця

Основні функції	**Варіанти реалізації**	**Переваги**	**Недоліки**
F1	*А*	Висока швидкодія, оптимізація під різні платформи	Висока вартість розробки, складність введення нових алгоритмів
	Б	Нижча вартість, відносна легкість проведення експериментів	Низька швидкодія
F2	А	Кросплатформність, полегшений процес завантаження нових даних	Складність при тестуванні
	Б	Нижча вартість та вища швидкість розробки	Більш вузький функіонал
F3	*А*	Велика кількість постачальників даних, більща щвидкодія підяас отримання даних	Нижча зручність в користуванні, потреьа в спеціалізованому модулі обробки

	Б	Зручний інтерфейс отримання даних	Висока вартість розробки та підтримки

На основі аналізу позитивно-негативної матриці робимо висновок, що при розробці програмного продукту деякі варіанти реалізації функцій варто відкинути, тому, що вони не відповідають поставленим перед програмним продуктом задачам. Ці варіанти відзначені у морфологічній карті.

Функція *F1:*

Оскільки обидві мови можуть бути використані для розробки і пропонують різні переваги, то слід розглянути обидва варіанти.

Функція *F2:*

Оскільки для даного продукту важливою є швидкість роботи, відкидаємо варіант а).

Функція *F3:*

Оскільки для даного продукту важливою є універсальність застосування та повність даних, використаємо варіант а) як єдиний можливий.

Таким чином, будемо розглядати такий варіант реалізації ПП:

1. F1а – F2б – F3а

2. F1б – F2б – F3а

Для оцінювання якості розглянутих функцій обрана система параметрів, описана нижче.

4.2 Обґрунтування системи параметрів ПП

4.2.1 Опис параметрів

На підставі даних про основні функції, що повинен реалізувати програмний продукт, вимог до нього, визначаються основні параметри виробу, що будуть використані для розрахунку коефіцієнта технічного рівня.

Для того, щоб охарактеризувати програмний продукт, будемо використовувати наступні параметри:

- *X1* – швидкодія мови програмування;
- *X2* – об’єм пам’яті для коректної роботи програми;
- *X3* – час виконання навчання;
- *X4* – потенційний об’єм програмного коду.

X1: Відображає швидкодію операцій залежно від обраної мови програмування.

X2: Відображає необхідний для збереження та обробки даних об’єм оперативної пам’яті пристрою.

X3: Відображає час, який витрачається на навчання моделі з різними параметрами.

X4: Показує розмір програмного коду, який необхідно створити розробнику.

4.2.2 Кількісна оцінка параметрів

Головна функція F_0 – розробка програмного продукту, який розпізнає стиль картин за вхідними даними у вигляді зображення та виводить конкретний стиль з 70 впроваджених. На підставі даних про основні функції, що повинен реалізувати програмний продукт, вимог до нього, визначаються основні параметри виробу, що будуть використані для розрахунку коефіцієнта технічного рівня. На основі аналізу позитивно-негативної матриці робимо висновок, що при розробці програмного продукту деякі варіанти реалізації функцій варто відкинути, тому, що вони не відповідають поставленим перед програмним

продуктом задачам. Гірші, середні і кращі значення параметрів вибираються на основі вимог замовника й умов, що характеризують експлуатацію ПП як показано у табл. 4.2.

Таблиця 4.2 – Основні параметри ПП

Назва Параметра	**Умовні позначення**	**Одиниці виміру**	**Значення параметра**		
			гірші	**середні**	**кращі**
Швидкодія мови програмування	X1	нс/Оп	300	100	1
Об’єм пам’яті для коректної роботи	X2	Гб	4	2	1
Час навчання моделі та її розмітки	X3	с	30000	23400	11000
Потенційний об’єм програмного коду	X4	кількість рядків коду	2000	1300	800

За даними таблиці 4.2 будуються графічні характеристики параметрів – рис. 4.2 – рис. 4.5.

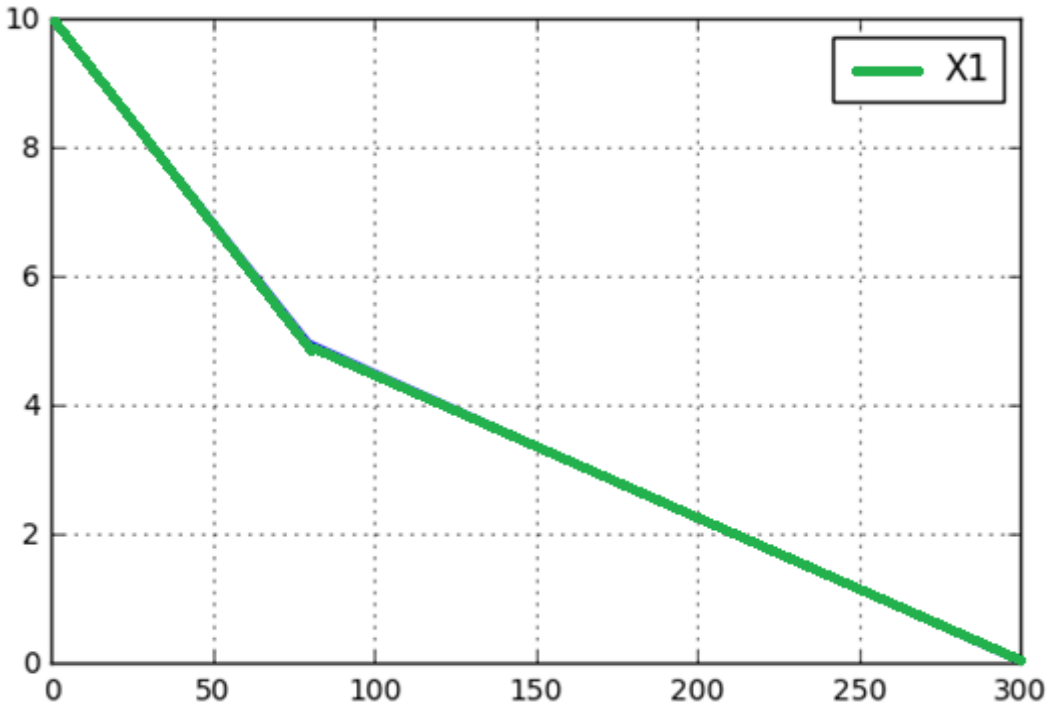


Рисунок 4.2 – X1, швидкодія мови програмування

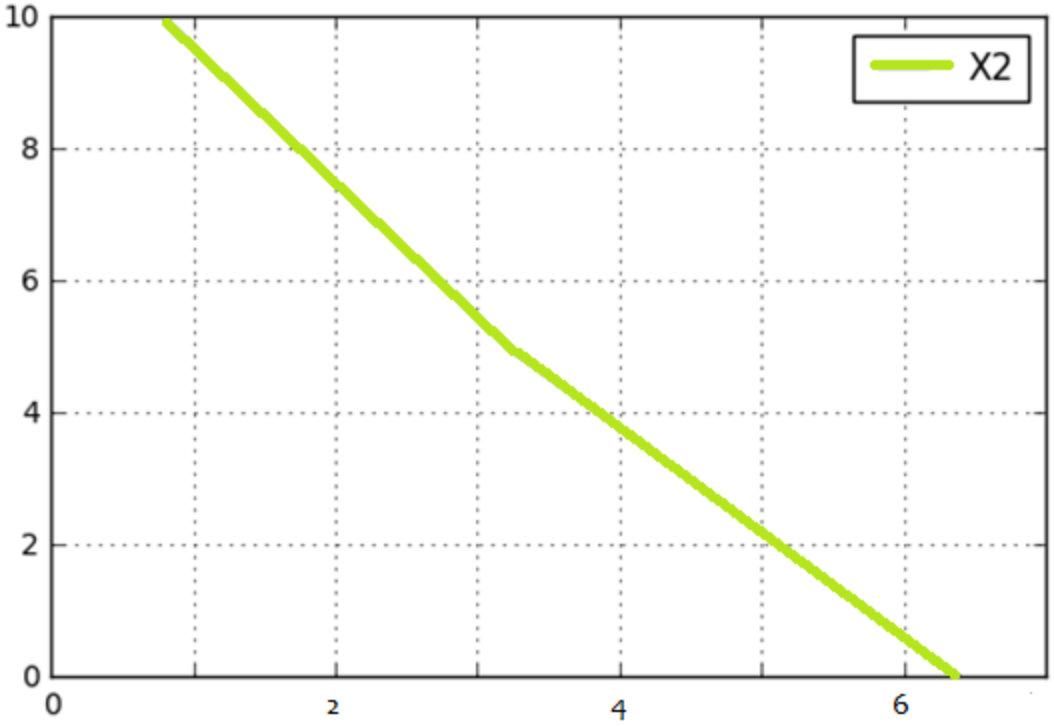


Рисунок 4.3 – X2, об’єм пам’яті для коректної роботи

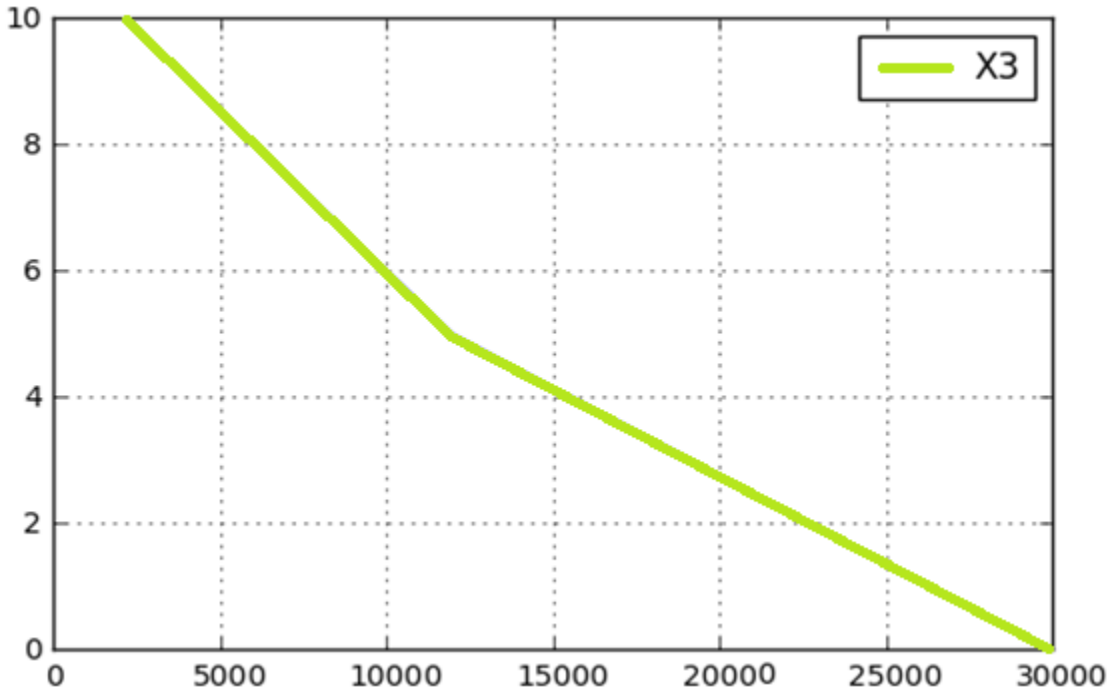


Рисунок 4.4 – X3, час виконання навчання

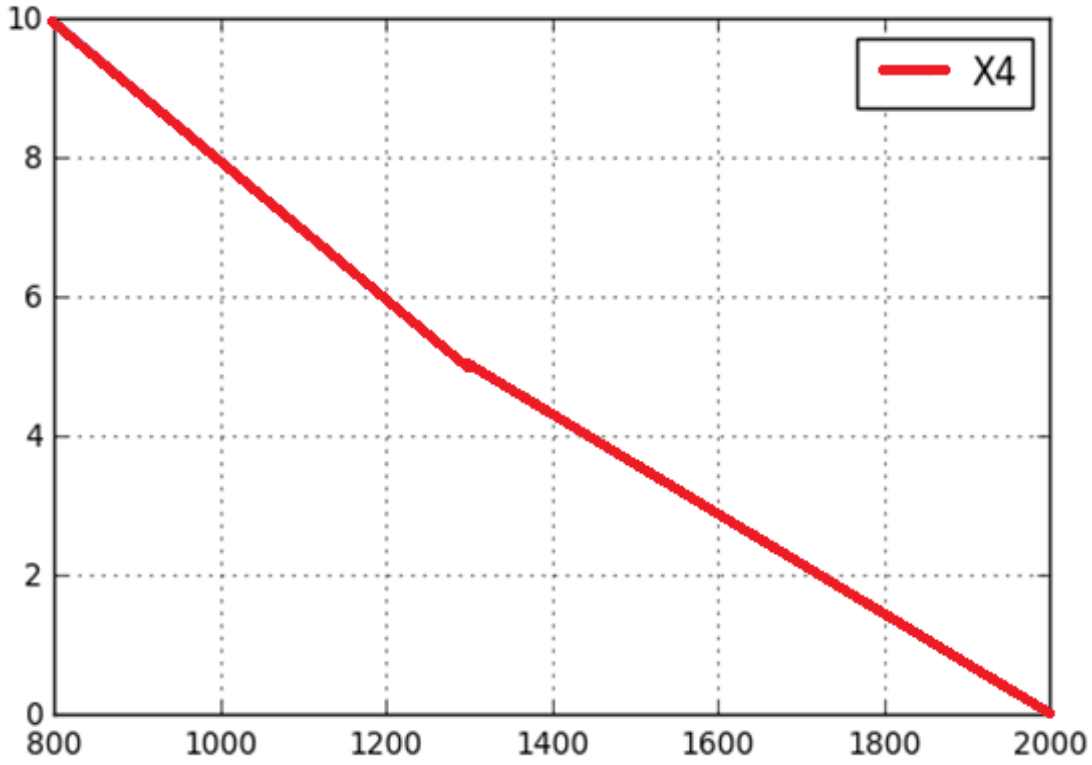


Рисунок 4.5 – X4, потенційний об’єм програмного коду

4.2.3 Аналіз експертного оцінювання параметрів

Після детального обговорення й аналізу кожний експерт оцінює ступінь важливості кожного параметру для конкретно поставленої цілі – розробка програмного продукту, який дає точні результати при знаходженні параметрів,а отже і найбільший потенціальний прибуток, моделей адаптивного прийняття рішень.

Значимість кожного параметра визначається методом попарного порівняння. Оцінку проводить експертна комісія із 7 людей. Визначення коефіцієнтів значимості передбачає:

- визначення рівня значимості параметра шляхом присвоєння різних рангів;
- перевірку придатності експертних оцінок для подальшого використання;
- визначення оцінки попарного пріоритету параметрів;
- обробку результатів та визначення коефіцієнту значимості.

Результати експертного ранжування наведені у таблиці 4.3.

Таблиця 4.3 – Результати ранжування параметрів

Познач. параметра	Назва параметра	Одиниці виміру	Ранг параметра за оцінкою експерта							Сума рангів R_i	Відхи-лення Δ_i	Δ_i^2
			1	2	3	4	5	6	7			
X1	Швидкодія мови програмування	нс/Оп	1	2	2	1	1	1	2	10	-7,5	56,25

X2	Об’єм пам’яті для коректної роботи	Мб	3	3	1	3	2	2	3	17	-0,5	0,25
X3	Час виконання програми	Мс	2	1	3	2	3	3	1	15	-2,5	6,25
X4	Потенційний об’єм програмного коду	к-сть рядків коду	4	4	4	4	4	4	4	28	10,5	110,25
	Разом		10	10	10	10	10	10	10	70	0	173

Для перевірки степені достовірності експертних оцінок, визначимо наступні параметри:

а) сума рангів кожного з параметрів і загальна сума рангів:

$$R_i = \sum_{i=1}^{N} r_{ij} R_{ij} = \frac{Nn(n+1)}{2} = 70$$

де N – число експертів, n – кількість параметрів;

б) середня сума рангів:

$$T = \frac{\mathrm{R_{ij}}}{\mathrm{n}} = 17{,}5.$$

в) відхилення суми рангів кожного параметра від середньої суми рангів:

$$\Delta_i = R_i - T$$

Сума відхилень по всім параметрам повинна дорівнювати 0;

г)загальна сума квадратів відхилення:

$$S = \sum_{i=1}^{N} \Delta_i^2 = 173.$$

Порахуємо коефіцієнт узгодженості:

$$W = \frac{12S}{N^2(n^3 - n)} = \frac{12 \cdot 173}{7^2(4^3 - 4)} = 0{,}706 > W_k = 0{,}67$$

Рангування можна вважати достовірним, тому що знайдений коефіцієнт узгодженості перевищує нормативний, який дорівнює 0,67.

Скориставшись результатами рангування, проведемо попарне порівняння всіх параметрів і результати занесемо у таблицю 4.4.

Таблиця 4.4 – Попарне порівняння параметрів

Параметри	Експерти							Кінцева оцінка	Числове значення
	1	2	3	4	5	6	7		
X1 і X2	<	<	>	<	<	<	<	<	0,5
X1 і X3	<	>	<	<	<	<	>	<	0,5
X1 і X4	<	>	<	<	<	<	<	<	0,5
X2 і X3	>	>	>	>	<	<	>	>	1,5
X2 і X4	<	<	<	<	<	<	<	<	0,5
X3 і X4	<	<	<	<	<	<	<	<	0,5

Числове значення, що визначає ступінь переваги *i*–го параметра над *j*–тим, a_{ij} визначається по формулі:

$$a_{ij} = \begin{cases} 1{,}5 \text{ при } X_i > X_j \\ 1.0 \text{ при } X_i = X_j \\ 0.5 \text{ при } X_i < X_j \end{cases}$$

З отриманих числових оцінок переваги складемо матрицю А=$\| a_{ij} \|$.

Для кожного параметра зробимо розрахунок вагомості $K_{вi}$ за наступними формулами:

$$K_{\text{вi}} = \frac{b_i}{\sum_{i=1}^{n} b_i}, \text{ де } b_i = \sum_{i=1}^{N} a_{ij}.$$

Відносні оцінки розраховуються декілька разів доти, поки наступні значення не будуть незначно відрізнятися від попередніх (менше 2%).На другому і наступних кроках відносні оцінки розраховуються за наступними формулами:

$$K_{\text{вi}} = \frac{b_i'}{\sum_{i=1}^{n} b_i'}, \text{де } b_i' = \sum_{i=1}^{N} a_{ij} b_j.$$

Як видно з таблиці 4.5, різниця значень коефіцієнтів вагомості не перевищує 4,6%, тому більшої кількості ітерацій не потрібно.

Таблиця 4.5 – Розрахунок вагомості параметрів

Параметриx_i	Параметриx_j				Перша ітер.		Друга ітер.		Десята ітер	
	X1	X2	X3	X4	b_i	$K_{\text{вi}}$	b_i^1	$K_{\text{вi}}^1$	b_i^2	$K_{\text{вi}}^2$

X1	1	1,5	1,5	1,5	5,5	0,323529	30,25	1	25329516	0,787874
X2	0,5	1	1,5	1,5	4,5	0,264706	20,25	0,5	3405063	0,105914
X3	0,5	1,5	1	1,5	4,5	0,264706	20,25	0,5	3405063	0,105914
X4	0,5	0,5	0,5	1	2,5	0,147059	6,25	0,5	9536,743	0,000297
Всього:					17	1	77	1	32149179	1

4.3 Аналіз рівня якості варіантів реалізації функцій

Визначаємо рівень якості кожного варіанту виконання основних функцій окремо.

Абсолютні значення параметрів *X2*(об'єм необхідної оперативної пам'яті) та *X3* (час обробки зображення) відповідають технічним вимогам умов функціонування даного ПП.

Абсолютне значення параметра *X1 (*швидкодія мови програмування*)* обрано не найгіршим (не максимальним), тобто це значення відповідає або варіанту а) 50 нс/Оп або варіанту б) 1 нс/Оп.

Коефіцієнт технічного рівня для кожного варіанта реалізації ПП розраховується так (таблиця 4.6):

$$K_K(j) = \sum_{i=1}^{n} K_{вi,j} B_{i,j} ,$$

де n – кількість параметрів;$K_{вi}$– коефіцієнт вагомості i–го параметра;B_i – оцінка i–го параметра в балах.

Таблиця 4.6 – Розрахунок показників рівня якості варіантів реалізації основних функцій ПП

Основні функції	Варіант реалізації функції	Абсолютне значення параметра	Бальна оцінка параметра	Коефіцієнт вагомості параметра	Коефіцієнт рівня якості
F1(X1)	А	1200	1,42	0,788	1,125714
F2(X2)	А	4	2,43	0,11	0,267143
F3(X3,X4)	А	800	2,14	0,11	0,235714
	Б	4000	4	0,00023	0,00092

За даними з таблиці 4.6 за формулою

$$K_K = K_{\text{ТУ}}[F_{1k}] + K_{\text{ТУ}}[F_{2k}] + \ldots + K_{\text{ТУ}}[F_{zk}],$$

визначаємо рівень якості кожного з варіантів:

K_{K1} = 1,125714+ 0,267143+ 0,235714= 1,628571286

K_{K2} = 1,125714+ 0,267143+ 0,00092= 1,393777

Як видно з розрахунків, кращим є перший варіант, для якого коефіцієнт технічного рівня має найбільше значення.

4.4 Економічний аналіз варіантів розробки ПП

Для визначення вартості розробки ПП спочатку проведемо розрахунок трудомісткості.

Всі варіанти включають в себе два окремих завдання:

1. Розробка проекту програмного продукту;

2. Розробка програмної оболонки;

Завдання 1 за ступенем новизни відноситься до групи А, завдання 2 – до групи Б. За складністю алгоритми, які використовуються в завданні 1 належать до групи 1; а в завданні 2 – до групи 3.

Для реалізації завдання 1 використовується довідкова інформація, а завдання 2 використовує інформацію у вигляді даних.

Проведемо розрахунок норм часу на розробку та програмування для кожного з завдань.

Проведемо розрахунок норм часу на розробку та програмування для кожного з завдань. Загальна трудомісткість обчислюється як

$$T_O = T_P \cdot K_П \cdot K_{СК} \cdot K_M \cdot K_{СТ} \cdot K_{СТ.М}, \qquad (5.1)$$

де T_P – трудомісткість розробки ПП;$K_П$ – поправочний коефіцієнт; $K_{СК}$ – коефіцієнт на складність вхідної інформації; K_M – коефіцієнт рівня мови програмування; $K_{СТ}$ – коефіцієнт використання стандартних модулів і прикладних програм; $K_{СТ.М}$ – коефіцієнт стандартного математичного забезпечення

Для першого завдання, виходячи із норм часу для завдань розрахункового характеру степеню новизни А та групи складності алгоритму 1, трудомісткість дорівнює: T_P = 90 людино-днів. Поправочний коефіцієнт, який враховує вид нормативно-довідкової інформації для першого завдання: $K_П$ = 1.6. Поправочний коефіцієнт, який враховує складність контролю вхідної та вихідної інформації для всіх семи завдань рівний 1: $K_{СК}$ = 1. Оскільки при розробці першого завдання використовуються стандартні модулі, врахуємо це за допомогою коефіцієнта $K_{СТ}$ = 0.8. Тоді, за формулою 5.1, загальна трудомісткість програмування першого завдання дорівнює:

$T_1 = 90 \cdot 1.6 \cdot 0.8 = 112,2$ людино-днів.

Проведемо аналогічні розрахунки для подальших завдань.

Для другого завдання (використовується алгоритм третьої групи складності, степінь новизни Б), тобто $T_Р = 28$ людино-днів, $К_П = 0.7$,$К_{СК} = 1$,$К_{СТ} = 0,8$:

$$T_2 = 28 \cdot 0,7 \cdot 0,8 = 15,68 \text{ людино-днів.}$$

Складаємо трудомісткість відповідних завдань для кожного з обраних варіантів реалізації програми, щоб отримати їх трудомісткість:

$$T_I = (112,2 + 15,68 + 4,8 + 15,68) \cdot 8 = 1186,88 \text{ людино-годин;}$$
$$T_{II} = (112,2 + 15,68 + 6,91 + 15,68) \cdot 8 = 1203,76 \text{ людино-годин;}$$

Найбільш високу трудомісткість має варіант II.

В розробці бере участь один фінансовий інженер, що спеціалізується на машинному навчанні, з окладом 28000 грн. Визначимо зарплату за годину за формулою:

$$С_ч = \frac{М}{T_m \cdot t} \text{ грн.,}$$

де М – місячний оклад працівників;T_m – кількість робочих днів тиждень;t – кількість робочих годин в день.

$$СЧ = \frac{28000}{1 * 21 * 8} = 166{,}67$$

Тоді, розрахуємо заробітну плату за формулою

$$С_{ЗП} = С_{ч} \cdot Т_{i} \cdot К_{Д},$$

де $С_{ч}$– величина погодинної оплати праці програміста;$Т_{i}$ – трудомісткість відповідного завдання; $К_{Д}$ – норматив, який враховує додаткову заробітну плату.

Зарплата розробників становить:

I. $С_{ЗП} = 166{,}67 \cdot 1186{,}88 \cdot 1{,}2 = 237376$

II. $С_{ЗП} = 166{,}67 \cdot 1203{,}76 \cdot 1{,}2 = 240752$

Відрахування на соціальний внесок становить 22,0%:

I. $С_{ВІД} = С_{ЗП} \cdot 0{,}22 = 237376 \cdot 0{,}22 = 52222{,}72$

II. $С_{ВІД} = С_{ЗП} \cdot 0{,}22 = 240752 \cdot 0{,}22 = 52965{,}44$

Тепер визначимо витрати на оплату однієї машино-години. ($С_{М}$)

Так як одна ЕОМ обслуговує одного інженера з окладом 28000 грн., з коефіцієнтом зайнятості 0,2 то для однієї машини отримаємо:

$С_{Г} = 12 \cdot М \cdot К_{З} = 12 \cdot 28000 \cdot 0{,}2 = 67200$ грн.

З урахуванням додаткової заробітної плати:

$С_{ЗП} = С_{Г} \cdot (1 + К_{З}) = 67200 \cdot (1 + 0.2) = 80640$ грн.

Відрахування на соціальний внесок:

$С_{ВІД}= С_{ЗП} \cdot 0.22 = 80640 * 0,22 = 17740,8$грн.

Амортизаційні відрахування розраховуємо при амортизації 25% та вартості ЕОМ – 140000 грн.

$С_А = К_{ТМ} \cdot К_А \cdot Ц_{ПР} = 1.15 \cdot 0.25 \cdot 140000 = 40250$грн.,

де $К_{ТМ}$– коефіцієнт, який враховує витрати на транспортування та монтаж приладу у користувача; $К_А$– річна норма амортизації; $Ц_{ПР}$– договірна ціна приладу.

Витрати на ремонт та профілактику розраховуємо як:

$С_Р = К_{ТМ} \cdot Ц_{ПР} \cdot К_Р = 1.15 \cdot 140000 \cdot 0.05 = 8050$грн.,

де $К_Р$– відсоток витрат на поточні ремонти.

Ефективний годинний фонд часу ПК за рік розраховуємо за формулою:

$Т_{ЕФ} =(Д_К – Д_В – Д_С – Д_Р) \cdot t_З \cdot К_В = (365 – 104 – 8 – 16) \cdot 8 \cdot 0.9 = 1706,4$ годин,

де $Д_К$ – календарна кількість днів у році; $Д_В$, $Д_С$ – відповідно кількість вихідних та святкових днів; $Д_Р$ – кількість днів планових ремонтів устаткування; t –кількість робочих годин в день; $К_В$– коефіцієнт використання приладу у часі протягом зміни.

Витрати на оплату електроенергії розраховуємо за формулою:

$$С_{ЕЛ} = Т_{ЕФ} \cdot N_С \cdot К_З \cdot Ц_{ЕН} = 1706,4 \cdot 0,22 \cdot 0,78 \cdot 2,7515 = 805,69 грн.,$$

де $N_С$ – середньо-споживча потужність приладу; $К_З$– коефіцієнтом зайнятості приладу; $Ц_{ЕН}$ – тариф за 1 КВт-годин електроенергії.

Накладні витрати розраховуємо за формулою:

$$С_Н = Ц_{ПР} \cdot 0.67 = 140000 \cdot 0,67 = 93800 грн.$$

Тоді, річні експлуатаційні витрати будуть:

$$С_{ЕКС} = С_{ЗП} + С_{ВІД} + С_А + С_Р + С_{ЕЛ} + С_Н$$

$$С_{ЕКС} = 80640 + 17740,8 + 40250 + 8050 + 805,69 + 93800 = 241286,49 грн.$$

Собівартість однієї машино-години ЕОМ дорівнюватиме:

$$С_{М\text{-}Г} = С_{ЕКС} / Т_{ЕФ} = 241286,49 / 1706,4 = 141,41 грн/час.$$

Оскільки в даному випадку всі роботи, які пов‘язані з розробкою програмного продукту ведуться на ЕОМ, витрати на оплату машинного часу складають:

$$С_М = С_{М\text{-}Г} \cdot Т$$

I. $С_М = 141,41 \cdot 1186,88 = 167825,90$ грн.

II. $С_М = 141,41 \cdot 1203,76 = 170223,71$ грн.

Накладні витрати складають 67% від заробітної плати:

$$С_Н = С_{ЗП} \cdot 0,67$$

I. $C_Н = 167825{,}90 \cdot 0{,}67 = 112443{,}36$грн.

II. $C_Н = 170223{,}71 \cdot 0{,}67 = 114049{,}89$ грн.

Отже, вартість розробки ПП становить:

$C_{ПП} = C_{ЗП} + C_{ВІД} + C_М + C_Н$

I. $C_{ПП} = 237376 + 52222{,}72 + 167825{,}90 + 112443{,}36 = 569867{,}98$ грн.

II. $C_{ПП} = 240752 + 52965{,}44 + 170223{,}71 + 114049{,}89 = 577991{,}04$ грн.

4.5 Вибір кращого варіанта ПП техніко-економічного рівня

Розрахуємо коефіцієнт техніко-економічного рівня за формулою:

$$К_{ТЕРj} = К_{Кj} / С_{Фj},$$

$К_{ТЕР1} = 1{,}628571286 / 569867{,}98 = 2{,}857 \cdot 10^{-6}$;

$К_{ТЕР2} = 1{,}393777 / 577991{,}04 = 2{,}411 \cdot 10^{-6}$;

Як бачимо, найбільш ефективним є перший варіант реалізації програми з коефіцієнтом техніко-економічного рівня $К_{ТЕР1} = 2{,}857 \cdot 10^{-6}$.

4.6 Висновки до розділу 4

В даному розділі проведено повний функціонально-вартісний аналіз ПП, який було розроблено в рамках дипломного проекту. Процес аналізу можна умовно розділити на дві частини.

В першій з них проведено дослідження ПП з технічної точки зору: було визначено основні функції ПП та сформовано множину варіантів їх реалізації; на основі обчислених значень параметрів, а також експертних оцінок їх важливості було обчислено коефіцієнт технічного рівня, який і дав змогу визначити оптимальну з технічної точки зору альтернативу реалізації функцій ПП.

Другу частину ФВА присвячено вибору із альтернативних варіантів реалізації найбільш економічно обґрунтованого. Порівняння запропонованих варіантів реалізації в рамках даної частини виконувалось за коефіцієнтом ефективності, для обчислення якого були обчислені такі допоміжні параметри, як трудомісткість, витрати на заробітну плату, накладні витрати.

Після виконання функціонально-вартісного аналізу програмного комплексу що розроблюється, можна зробити висновок, що з альтернатив, що залишились після першого відбору двох варіантів виконання програмного комплексу оптимальним є перший варіант реалізації програмного продукту. У нього виявився найкращий показник техніко-економічного рівня якості $К_{ТЕР} = 6{,}2 \cdot 10^{-6}$.

Цей варіант реалізації програмного продукту має такі параметри:

- мова програмування – Python;
- бібліотека бектестингу Pandas;
- постачальник економічних даних EXMO.

Даний варіант виконання програмного комплексу дає користувачу можливість провидити трейдерські операції на ринку криптовалют, отрімувати актуальну інформацію зо до його курсу, вікористовувати адаптивну систему для автоматичної торгівлі чи супроводу прийняття рішень.

ВИСНОВКИ

У даній дипломній роботі було детально розглянуто ефектуаційний підхід до проведення операцій на ринку цінних паперів, на основі якого сформовано адаптивний метод прийняття рішень. Також було розглянуто можливі засоби його реалізації та проаналізовано їх роботу.

Для розв’язку задачі було проведено порівняльний аналіз методів машинного навчання. Відповідно до концепції ефектуації обрано актив для реалізації методу. В процесі розробки методу прийняття рішень було показано ефективність моделей прийняття рішень без використання прогнозів поведінки ринку , важливість “пам’яті” для алгоритму. Також було досліджено особливості обраного ринку крипто валют, зокрема виявлено атракторність сукупної поведінки гравців та добові ритми системи. Виявлено приховані коригуючі операції біржі.

Також було проаналізовано основні етапи створення моделі та її параметри з їх впливом на систему. В результаті роботи отримано модель що дозволяє мало ризиковано отримувати прибуток за низького початкового капіталу. Отримані результати свідчать досягнення прибутковості моделі на рівні середньої банківської ставки (8% в місяць).

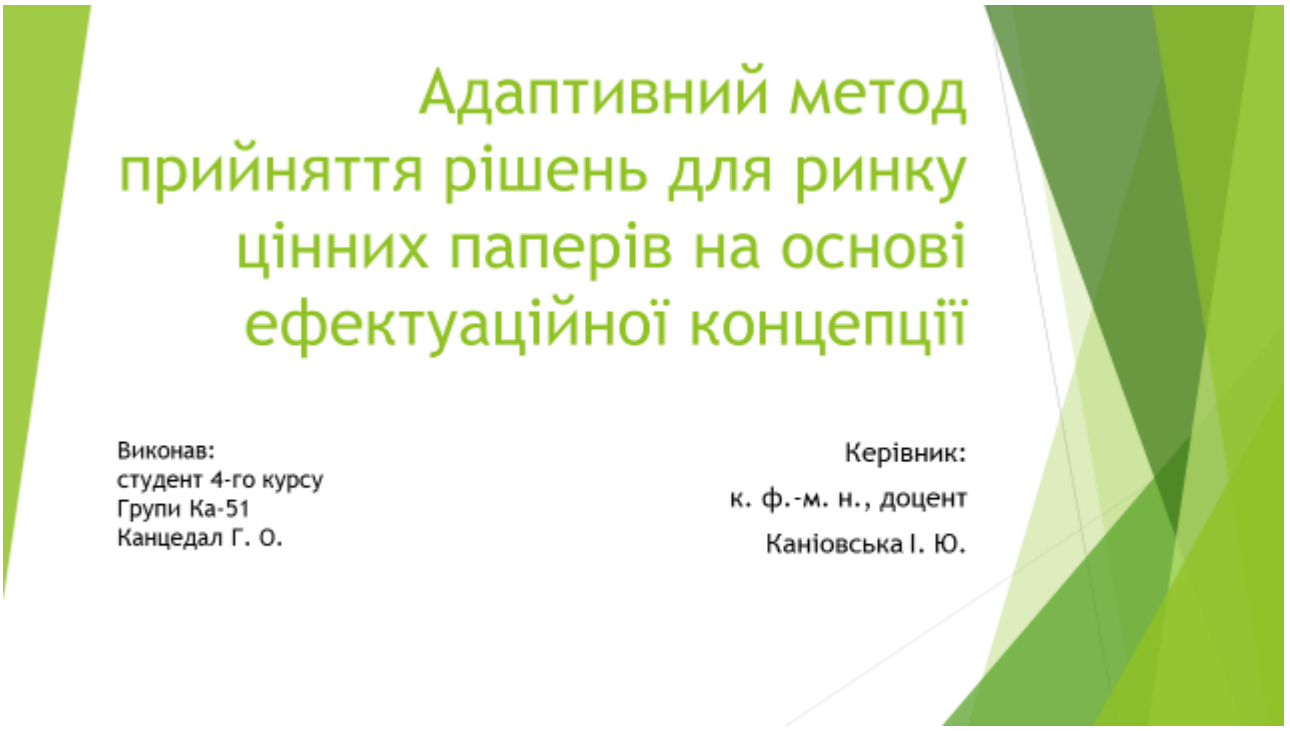
У подальшому дослідженні планується розширити кількість валютних пар, провести тестування моделі з більшою частотою дискретизації з комбінуванням декількох бірж та дослідити крос парну торгівлю.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Sarasvathy S. D., Kotha S. Effectuation in the Management of Knightian Uncertainty: Evidence from the Real Networks Case : *Research on Management and Entrepreneurship*, 2001. 55 p.
2. Dew N., Sarasvathy S. D., Read S., Wiltbank R. Affordable Loss: Behavioral Economic Aspects of the Plunge: *Strategic Entrepreneurship*, 2009. 27 p.
3. Dew N., Sarasvathy S. D. New Market Creation through Transformation: *Journal of Evolutionary Economics*. 2005. 27 p.
4. Wiltbank R., Dew N., Read S., Sarasvathy S. D. What to Do Next? The Case for Non-Predictive Strategy: *Strategic Management Journal*. 2006. 98 p.
5. Rob Iati The Real Story of Trading Software Espionage: *The TABB Group*, 2009. 85 p.
6. Cohen, J., Cohen P., West, S.G. & Aiken, L.S. Applied Multiple Regression/Correlation Analysis for the Behavioral Sciences: *Strategic Management Journal*, 2002. 76p.
7. Li, Ping, Qiang Wu, and Christopher J. Burges. Mcrank: Learning to rank using multiple classification and gradient boosting: *Advances in Neural Information Processing Systems*, 2007. 120 p.
8. Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, Tie-Yan Liu. LightGBM: A Highly Efficient Gradient Boosting Decision Tree: *Advances in Neural Information Processing Systems*, 2017. 252p.
9. Shi, Haijian. Best-first decision tree learning: *The University of Waikato*, 2007.
10. Friedman, J. H. Greedy Function Approximation: *A Gradient Boosting Machine*, 1999. 78p.

11. Shafer J., Rakesh A., Manish M. SPRINT: A scalable parallel classifier for data mining: *Very Large Data Bases*, 1996. 13p.
12. Thakur R., Rolf R., Gropp W. Optimization of collective communication operations in MPICH: *International Journal of High Performance Computing Applications*, 2005. 123p.

ДОДАТОК А Ілюстративні матеріали для доповіді



Актуальність дослідження

- Задача прийняття рішень на ринках цінних паперів є провідною для сучасного світу.
- Побудова стратегій з низьким ступенем ризику є оптимальним вибором для збереження пасивів будь-якого підприємства.
- Побудована модель може бути легко перенесена на будь-який ринок, в основі якого лежить часовий ряд та контрагентами ряду виступають рівноправні користувачі без значної переваги.

- Об'єкт дослідження

Фінансові часові ряди цінних паперів, історія торгівельних операцій.

- **Предмет дослідження**

Методи класифікації, що базуються на машинному навчанні.

- **Мета дослідження**

Проаналізувати предмет дослідження, реалізувати безперервний доступ до трейдингових даних, провести тестування побудованої моделі на історичних даних та в режимі реального часу.

Постановка задачі

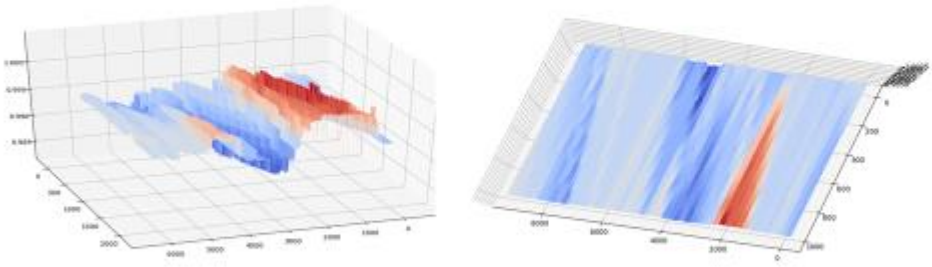
- Розробити підходи для безпосереднього доступу до даних торгової біржі, накопити історичні данні для попереднього аналізу.
- Обрати відповідну торгівельну пару для подальшої розробки та сформувати критерій успішної роботи системи.
- Сформувати правила, що відтворюють торгівельні операції на біржі та протестувати отриману систему в реальному часі.

Формалізація задачі

- Поставлена задача була зведена до задачі класифікації у n-вимірному просторі класу точок, що розподіляються на 3 групи – нейтральні, закупівлі, продаж.
- Побудовано моделі, що ґрунтуються на 113 вимірах (несуть інформацію за відповідний момент часу),155 вимірах(відомості про досліджуваний момент та зміни його порівняно з попереднім) та 255 (інформація за останні 2 виміри та їх зміни).
- Подальше введення інших не дало покращення результатів.

Формалізація задачі

- В результаті формалізації отримали наступні результати (по осі [0,6000] - час купівлі активу [0,2000] - час утримання активу) Точки що мають темно-коричневий колір були класифіковані як час купівлі, а відповідна їм ордината як час продажу



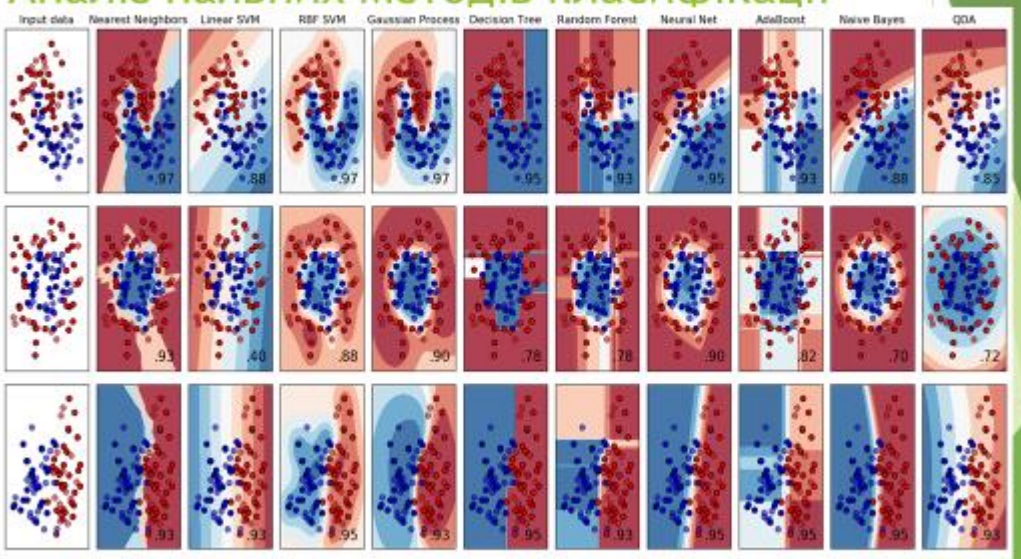
Аналіз наявних методів класифікації

- Для вирішення задачі були розглянуті наступні методи: Метод найближчих сусідів;Лінійний SVM метод; RBF SVM; Метод побудований на гаусівських процесах; Дерево прийняття рішень; Випадковий ліс (комбінація дерев прийняття рішень); Нейронні мережі; ADa boost; Баєсівські мережі; QDA.
- Найгірші результати, з позиції довгострокового прогнозування має метод AdaBoost, оскільки має білі області, що у випадку класифікації дасть велику групу точок, що не будуть віднесені до жодного з класів і відразу стануть похибкою моделі(причому частина тренувальних точок потрапила в ці області, що зовсім погано).

Аналіз наявних методів класифікації

- Методи Decidion tree та Random forist базуються на алгоритмі, що дозволяє розділяти данні в 2 мірному просторі лише горизонтальними та вертикальними лініями (точніше Random forist базується на Decidion tree). Однак в методі Random forist спостерігається перенавчання: велика кількість невеликих областей з високою імовірністю (з однією тренувальною точкою) та значна кількість областей з відсутньою характеристикою.
- Лінійний SVM метод показав себе повністю не пристосованим до поставленої задачі.
- Методи найближчих сусідів, RBF SVM, гаусівський процес показали непогані результати однак потребували значно більшу кількість обчислювальних ресурсів і в по загальному відхиленню не надто відрізнялись від дерев прийняття рішень.

Аналіз наявних методів класифікації

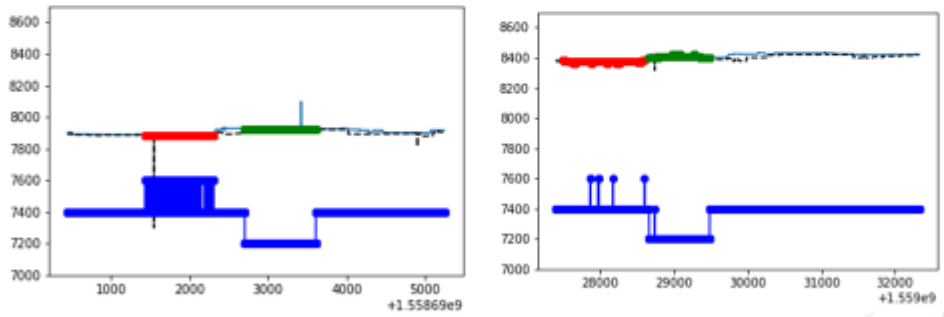


Результати аналізу

- Обрано технологію gradient boosting на базу дерев рішень (а точніше модифікацію LGBM).
- Основною відмінністю від інших алгоритмів побудованих на деревах рішень є вертикальне (leaf-wise -методологія нарощування листків і вузлів дерева, що орієнована на ефективну побудову усього дерева) нарощування листків дерева

Робота моделі

Робота класифікатора побудованого на основы технології LGBM



Робота класифікатора на навчальних даних та тестових даних

Висновки

- Модель побудована на базі ефектуаційної концепції дозволяє отримувати прибуток з низького початкового капіталу з невисоким ступенем ризику.
- Отримані результати свідчать досягнення прибутковості моделі на рівні середньої банківської ставки : 8% в місяць.
- Досягнуто адаптивності алгоритму для будь-якого активу та можливість довгої автономної роботи за рахунок навчання моделі
- За допомогою моделі було виявлено приховані закономірності на ринку крипто валют.

Подальші дослідження

- Переглянути більшу кількість валютних пар, розглянути можливості між біржової торгівлі.
- Залучити більшу кількість параметрів до моделі
- Провести тестування моделі з більшою дискретизацією та на більших проміжках часу, виявити оптимальний час роботи моделі без навчання.

Дякую за увагу!

ДОДАТОК Б Лістинг програми

```
PAIRS = ["BTC_USD","BTC_UAH", "BTC_EUR",
        "ETH_USD", "ETH_UAH", "ETH_EUR","ETH_BTC"]
ANSVERS = ['ask_quantity', 'ask_amount', 'ask_top', 'bid_quantity',
         'bid_amount', 'bid_top']
GLASS = ['ask', 'bid']
G_PARAM = ['mean_ammount', 'spred', 'more_mean_num', 'first', 'top_amount']
import pandas as pd
import numpy as np
#GOD const
MAX_WAIT_TIME = 5
TIME_FOR_QUERY = 0.3
import sys
import http.client
import urllib
import json
import hashlib
import hmac
import time
class ExmoAPI:
    def __init__(self, API_KEY, API_SECRET, API_URL = 'api.exmo.com',
API_VERSION = 'v1'):
        self.API_URL = API_URL
        self.API_VERSION = API_VERSION
        self.API_KEY = API_KEY
        self.API_SECRET = bytes(API_SECRET, encoding='utf-8')
```

```
        self.LUST_TIME = 0

    def sha512(self, data):
        H = hmac.new(key = self.API_SECRET, digestmod = hashlib.sha512)
        H.update(data.encode('utf-8'))
        return H.hexdigest()

    def api_query(self, api_method, params = {}, typ = "POST"):
        params['nonce'] = int(round(time.time() * 1000))
        params =  urllib.parse.urlencode(params)

        self.LUST_TIME = time.time()
        sign = self.sha512(params)
        headers = {
            "Content-type": "application/x-www-form-urlencoded",
            "Key": self.API_KEY,
            "Sign": sign
        }
        conn = http.client.HTTPSConnection(self.API_URL)
        conn.request(typ, "/" + self.API_VERSION + "/" + api_method, params,
headers)
        response = conn.getresponse().read()

        conn.close()

        try:
            obj = json.loads(response.decode('utf-8'))
            if 'error' in obj and obj['error']:
```

```
                print(obj['error'])
                return "error"
                #raise sys.exit()
            return obj
        except json.decoder.JSONDecodeError:
            print('Error while parsing response:', response)
            return "error"
            #raise sys.exit()

    def safe_query(self, api_method, params = {},typ = "POST"):
        if time.time() - self.LUST_TIME < TIME_FOR_QUERY:
            time.sleep(time.time() - self.LUST_TIME)

        ret = self.api_query( api_method, params, typ)
        start = time.time()
        while((ret == "error")&(time.time() - start<MAX_WAIT_TIME)):
            if time.time() - self.LUST_TIME < TIME_FOR_QUERY+0.02:
                time.sleep(time.time() - self.LUST_TIME+0.02)
            ret = self.api_query( api_method, params)
        return ret
        ExmoAPI_instance = ExmoAPI(API_key, API_secret)
print(ExmoAPI_instance.api_query('user_info'))
def test_spead ():
    start = time.time()
    count = 0
    while (1):
        print(ExmoAPI_instance.safe_query('user_info')['server_date'])
        count += 1
```

```
        print((time.time() - start)/60)
        if (time.time() - start)/60 >1:
            print(count)
            count = 0
            start = time.time()
def col_gen():
    col = []
    col.append("TIME")
    for i in PAIRS:
        for j in ANSVERS:
            col.append(i+' '+j)
        for j in GLASS:
            for k in G_PARAM:
                col.append(i+' '+j+' '+k)
    return col
def get_curs_info(ExmoAPI_inst):
    reque =""
    for i in PAIRS:
        reque = reque+","+str(i)
    jsdata = ExmoAPI_inst.safe_query("order_book/?pair="+reque,typ = "GET")
    outdata = []
    outdata.append(round(time.time(),2))
    col = []
    col.append("TIME")
    for i in PAIRS:
        for j in ANSVERS:

            outdata.append(jsdata[i][j])
```

```
            col.append(i+' '+j)

        #['mean_ammount', 'spred', 'more_mean_num', 'first', 'top_amount']
        for j in GLASS:
            h = np.array(jsdata[i][j])
            h = np.float32(h)
            mean = np.mean(h[:,1])

            outdata.append(mean)
            col.append(i+' '+j+' '+'mean_ammount')

            outdata.append((abs(h[:,0][0]-h[:,0][-1])))
            col.append(i+' '+j+' '+'spred')

            outdata.append(np.size(np.where(h[:,1]>np.mean(h[:,1]))))
            col.append(i+' '+j+' '+'more_mean_num')

            outdata.append(h[:,0][np.where(h[:,1]>np.mean(h[:,1]))][0])
            col.append(i+' '+j+' '+'first')

            outdata.append(h[:,1][0])
            col.append(i+' '+j+' '+'top_amount')


    res = pd.DataFrame(index=np.arange(0, 1),columns=col)
    res.iloc[0] = outdata
    return res
col = col_gen()
```

```
        history = pd.DataFrame(columns=col)
        history
        i = 10000
        while i > 0:
            i -= 1
            print(i)
            history = pd.concat([history, get_curs_info(ExmoAPI_instance)],
ignore_index=True)
        history.to_csv("data6.csv")
        def takeClosest(myList, myNumber):
            """
            Assumes myList is sorted. Returns closest value to myNumber.

            If two numbers are equally close, return the smallest number.
            """
            pos = bisect_left(myList, myNumber)
            if pos == 0:
                return myList[0]
            if pos == len(myList):
                return myList[-1]
            before = myList[pos - 1]
            after = myList[pos]
            if after - myNumber < myNumber - before:
                return after, pos
            else:
                return before, pos -1
        #ANSVERS = ['ask_quantity', 'ask_amount', 'ask_top', 'bid_quantity',
'bid_amount', 'bid_top']
```

```
        #G_PARAM = ['mean_ammount', 'spred', 'more_mean_num', 'first', 'top_amount']
        def bbid(T, pair, amount, hist):    # Покупка
            i = takeClosest(history["TIME"],T)[1]+10
            price = hist.iloc[i][str(pair+' bid_top')]
            stopamount = hist.iloc[i][str(pair+" bid "+ G_PARAM[4])]
            if amount < stopamount:
                return -amount*price*(1+, + (1-commision)*amount, price
            return -stopamount*price, + (1-commision)*stopamount, price
        def aask(T, pair, amount, hist):
            i = takeClosest(history["TIME"],T)[1]+10
            price = hist.iloc[i][str(pair+' ask_top')]
            stopamount = hist.iloc[i][str(pair+" ask "+ G_PARAM[4])]
            if amount < stopamount:
                return +amount*price, + (1-commision)*amount, price
            return +stopamount*price, + (1-commision)*stopamount, price
        def trry(hist,i1,i2,wait, pair):
            i1 = int(round(i1))
            i2 = int(round(i2))
            if i2-i1>wait:
                return 0
            if i2>np.size(hist["TIME"]):
                return 0
            if i1<0:
                return 0
            #i1 = takeClosest(history["TIME"],T1)[1]
            price1 = hist.iloc[i1][str(pair+' ask_top')]
            #i2 = takeClosest(history["TIME"],T)[1]
```

```
        price2 = hist.iloc[i2][str(pair+' bid_top')]
        return price2 - price1
    def tt(x):
        return trry(history,x[0],x[1],x[2], "BTC_USD")
    buy = []
    sel = []
    waittime = 100
    i = 0
    for i in range(np.size(history["TIME"])-waittime-1):
        ddd=0
        print("start")
        good = 0
        for j in range(waittime):

            dd = float(history["BTC_USD ask_top"][i])*1.005 -
float(history["BTC_USD bid_top"][i+j])
            if (dd<0)&(round(dd, 3)!=ddd):
                good = i+j
                ddd=round(dd, 3)
                print("=)")
        if good!=0:
            buy.append(i)
            sel.append(good)
            print(good)
        i = i+waittime
    def takeClosest(myList, myNumber):
        """
        Assumes myList is sorted. Returns closest value to myNumber.
```

```
        If two numbers are equally close, return the smallest number.
        """
        pos = bisect_left(myList, myNumber)
        if pos == 0:
            return myList[0]
        if pos == len(myList):
            return myList[-1]
        before = myList[pos - 1]
        after = myList[pos]
        if after - myNumber < myNumber - before:
            return after, pos
        else:
            return before, pos -1
    def prepare(hist,waittime):
        buy = []
        sel = []
        for i in range(0, np.size(hist["TIME"])-waittime-1, 10):
            for j in range(1,waittime):
                dd = float(hist["BTC_USD bid_top"][i+j])/float(hist["BTC_USD ask_top"][i])*(1-commision)**2
                if dd>1:
                    buy.append(i)
                    sel.append(i+j)
        return buy,sel
    def make_plot(hist, b, s, pre, ite):
        plt.plot(hist["TIME"], hist["BTC_USD ask_top"], hist["TIME"], hist["BTC_USD bid_top"], '--k')
```

```
    plt.plot(hist["TIME"][b], hist["BTC_USD ask_top"][b], marker='o', color='r')
    plt.plot(hist["TIME"][s], hist["BTC_USD bid_top"][s], marker='o', color='g')
    plt.plot(hist["TIME"], pre*200+7400, marker='o', color='b')
    #plt.xlim(200000, 3000000000)
    plt.ylim(7000, 8700)
    plt.savefig("BTC_USD"+str(ite)+str(int(round(time.time(),-1))))
def serch(hist,time,mem):
    b,s = prepare(history,time)
    rez = np.zeros(np.size(hist["TIME"])-mem)
    for i in b:
        rez[i] = 1
    for i in s:
        rez[i] = -1
    return rez,b,s
history = pd.read_csv("dataday1559032330.csv")
train,b,s = serch(history,3000, 1)
model.fit(result, train)
pre = model.predict(result)
make_plot(history.drop(history.index[len(history)-1]), b, s, pre, 0)
model = xgboost.XGBClassifier()
for i in range(np.size(name)):
    history = pd.read_csv(name[i])
    b,s = prepare(history,1000)
    pred = np.zeros(np.size(history["TIME"]))
    for i in b:
        pred[i] = 1
    for i in s:
        pred[i] = -1
```

```
X = history
Y = pred
#seed = 7
#test_size = 0.33
#X_train, X_test, y_train, y_test = cross_validation.train_test_split(X, Y, test_size=test_size, random_state=seed)
model.fit(X, Y)
y_pred = model.predict(history)
make_plot(history, b, s, y_pred, str(i))
history = pd.read_csv(name[-1])
b,s = prepare(history,1000)
y_pred = model.predict(history)
make_plot(history, b, s, y_pred, str(str(i)+'future'))
```